Lecture Notes in Electrical Engineering 714

Erik Marchi · Sabato Marco Siniscalchi · Sandro Cumani · Valerio Mario Salerno · Haizhou Li *Editors*

Increasing Naturalness and Flexibility in Spoken Dialogue Interaction

10th International Workshop on Spoken Dialogue Systems

Springer

Lecture Notes in Electrical Engineering

Volume 714

The book series *Lecture Notes in Electrical Engineering* (LNEE) publishes the latest developments in Electrical Engineering - quickly, informally and in high quality. While original research reported in proceedings and monographs has traditionally formed the core of LNEE, we also encourage authors to submit books devoted to supporting student education and professional training in the various fields and applications areas of electrical engineering. The series cover classical and emerging topics concerning:

- Communication Engineering, Information Theory and Networks
- Electronics Engineering and Microelectronics
- Signal, Image and Speech Processing
- Wireless and Mobile Communication
- Circuits and Systems
- Energy Systems, Power Electronics and Electrical Machines
- Electro-optical Engineering
- Instrumentation Engineering
- Avionics Engineering
- Control Systems
- Internet-of-Things and Cybersecurity
- Biomedical Devices, MEMS and NEMS

For general information about this book series, comments or suggestions, please contact leontina.dicecco@springer.com.

To submit a proposal or request further information, please contact the Publishing Editor in your country:

China

Jasmine Dou, Editor (jasmine.dou@springer.com)

India, Japan, Rest of Asia

Swati Meherishi, Editorial Director (Swati.Meherishi@springer.com)

Southeast Asia, Australia, New Zealand

Ramesh Nath Premnath, Editor (ramesh.premnath@springernature.com)

USA, Canada:

Michael Luby, Senior Editor (michael.luby@springer.com)

All other Countries:

Leontina Di Cecco, Senior Editor (leontina.dicecco@springer.com)

**** This series is indexed by EI Compendex and Scopus databases. ****

More information about this series at http://www.springer.com/series/7818

Erik Marchi · Sabato Marco Siniscalchi ·
Sandro Cumani · Valerio Mario Salerno ·
Haizhou Li
Editors

Increasing Naturalness and Flexibility in Spoken Dialogue Interaction

10th International Workshop on Spoken Dialogue Systems

Springer

Editors
Erik Marchi
Apple
Cupertino, CA, USA

Sabato Marco Siniscalchi
Kore University of Enna
Enna, Italy

Sandro Cumani
Polytechnic University of Turin
Torino, Italy

Valerio Mario Salerno
Kore University of Enna
Enna, Italy

Haizhou Li
National University of Singapore
Singapore, Singapore

ISSN 1876-1100 ISSN 1876-1119 (electronic)
Lecture Notes in Electrical Engineering
ISBN 978-981-15-9322-2 ISBN 978-981-15-9323-9 (eBook)
https://doi.org/10.1007/978-981-15-9323-9

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

This book contains a selection of revised papers that were presented at the 10th edition of the International Workshop on Spoken Dialogue Systems (IWSDS) that took place in the beautiful town of Syracuse in Sicily (Italy), from 24 to 26 April 2019. IWSDS is usually held every year and provides a platform to present and discuss global research and application of spoken dialogue systems.

This 10th edition of IWSDS named "Increasing Naturalness and Flexibility in Spoken Dialogue Interaction" focused specifically on the following topics:

- Context Understanding and Dialogue Management
- Human–Robot Interaction
- Dialogue Evaluation and Analysis
- Chatbots and Conversational Agents
- Lifelong Learning
- Question Answering and other Dialogue Applications
- Dialogue Breakdown Detection

Cupertino, USA — Erik Marchi
Enna, Italy — Sabato Marco Siniscalchi
Torino, Italy — Sandro Cumani
Enna, Italy — Valerio Mario Salerno
Singapore, Singapore — Haizhou Li

Contents

Human-Robot Interaction

Dialogue Evaluation and Analysis

Chatbots and Conversational Agents

Lifelong Learning

Context Understanding and Dialogue Management

Skip Act Vectors: Integrating Dialogue Context into Sentence Embeddings

Jeremy Auguste, Frédéric Béchet, Géraldine Damnati, and Delphine Charlet

Abstract This paper compares several approaches for computing dialogue turn embeddings and evaluate their representation capacities in two dialogue act related tasks within a hierarchical Recurrent Neural Network architecture. These turn embeddings can be produced explicitely or implicitely by extracting the hidden layer of a model trained for a given task. We introduce *skip-act*, a new dialogue turn embeddings approach, which are extracted as the common representation layer from a multi-task model that predicts both the previous and the next dialogue act. The models used to learn turn embeddings are trained on a large dialogue corpus with light supervision, while the models used to predict dialog acts using turn embeddings are trained on a sub-corpus with gold dialogue act annotations. We compare their performances for predicting the current dialogue act as well as their ability to predict the next dialogue act, which is a more challenging task that can have several applicative impacts. With a better context representation, the *skip-act* turn embeddings are shown to outperform previous approaches both in terms of overall F-measure and in terms of macro-F1, showing regular improvements on the various dialogue acts.

J. Auguste (✉) · F. Béchet
Aix-Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France
e-mail: jeremy.auguste@lis-lab.fr

F. Béchet
e-mail: frederic.bechet@lis-lab.fr

G. Damnati · D. Charlet
Orange Labs, Lannion, France
e-mail: geraldine.damnati@orange.com

D. Charlet
e-mail: delphine.charlet@orange.com

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_1

1 Introduction

Following the successful application of continuous representation of words into vector spaces, or *embeddings*, in a large number of Natural Language Processing tasks [14, 15], many studies have proposed the same approach for larger units than words such as sentences, paragraphs or even documents [10, 11]. In all cases the main idea is to capture the *context of occurrence* of a given unit as well as the unit itself.

When processing dialog transcriptions, being able to model the *context of occurrence* of a given turn is of great practical use in applications such as automated dialog system for predicting the next action to perform, or analytics in order, for example, to pair questions and answers in a corpus of dialog logs. Therefore finding the best embedding representations for dialog turns in order to model dialog structure as well as the turns themselves is an active field of research.

In this paper, we evaluate different kinds of sentence-like (turns) embeddings on dialogue act classification tasks in order to measure how well they can capture dialog structures. In a first step, the dialogue turn embeddings are learned on large corpus of chat conversations, using a light supervision approach where dialogue act annotations are given by an automatic DA parser. Even if the annotations are noisy, this light supervision approach allows us to learn turn-level vector representations on a large amount of interactions. In a second step, the obtained turn-level vector representations are used to train dialogue act prediction models with a controlled supervised configuration.

After presenting the dialogue act parser architecture in Sect. 3, we will present the various dialogue turn embeddings approaches in Sect. 4. The corpus and the dialogue act annotation framework are presented in Sect. 5 while Sect. 6 describes the experimental results.

2 Related Work

In order to create and then evaluate the quality of embeddings, several different types of approaches have been proposed. For word embeddings, a lot of work has been done to try to evaluate how they are able to capture relatedness and similarity between two words by using manual annotation [4, 9, 12] and by using cognitive processes [2, 18]. However, on sentence embeddings, it is not easy to tell how similar or related two sentences are. Indeed, the context in which they appear is very important to truly understand the meaning of a sentence and how it interacts with other sentences.

Multiple papers propose different kinds of evaluation tasks in order to evaluate different kinds of sentence embeddings. In [8], the authors use the SICK [13] and STS 2014 [1] datasets to evaluate the similarity between sentences by using similarity ratings. They also use sentiment, opinion polarity and question type tasks to evaluate the embeddings. As these datasets are composed of sentence pairs without context, the proposed sentence embeddings approaches are only based on the sentence itself. In [7], sentence embeddings are evaluated by looking at their ability to capture surface, syntactic and semantic information. Here again, this framework primarily focuses on the sentence itself and not on the context in which it is produced. In [5], a sentence embeddings evaluation framework is proposed that groups together most of the previous evaluation tasks in addition to inference, captioning and paraphrase detection tasks. In all of the above approaches, the focus is on the evaluation of sentence embeddings such as Skip-thoughts [10], ParagraphVectors [11] or InferSent [6] in order to find out the embeddings that have the best properties in general. However, none of these embeddings and evaluation tasks are built to take into account dialogues and more specifically, the structure and interactions in a dialogue. Some work has been done in order to take into account the dialogue context in [17]. In their work, the authors try to take into account this context by using a modified version of word2vec to learn sentence embeddings on dialogues. These embeddings are then evaluated by comparing clusters of sentence embeddings with manually assigned dialogue acts. This allows to see if the learned embeddings capture information about the dialogue context, however it does not use explicit dialogue structure information to learn the embeddings. In our work, we use a corpus with a noisy dialogue act annotation to learn specialized sentence embeddings that try to directly capture information about the context and interactions in the dialogue.

3 Dialogue Act Parser Architecture

In order to be able to create sentence embeddings that take into account the dialogue context, we will be using dialogue acts. They allow us to partially represent the structure and the interactions in a dialogue. We use two different kinds of models to parse these dialogue acts where one kind is used to create sentence embeddings, while the second kind is used to later evaluate the different embeddings.

The first architecture is a 2-level hierarchical LSTM network where the first level is used to represent the turns in a conversation, and the second level represents the conversation, as shown in Fig. 1. The input is the sequence of turns which are themselves sequences of words represented as word embeddings. The word embeddings are trained by the network from scratch. The dialogue acts are predicted using the output for each turn at the second level. Since we do not use a bidirectional LSTM, the model only makes use of the associated turn and the previous turns of a conversation in order to predict a given act. It has no information about the future, nor about the previous acts. This architecture allows us to use the hidden outputs of the first layer as the sentence embeddings of each turn.

The second architecture is a simple LSTM network which only has a single layer, as shown in Fig. 2. The input sequence that is given to the LSTM is the sequence of turns of a conversation where each turn is replaced by a pre-trained turn embedding.

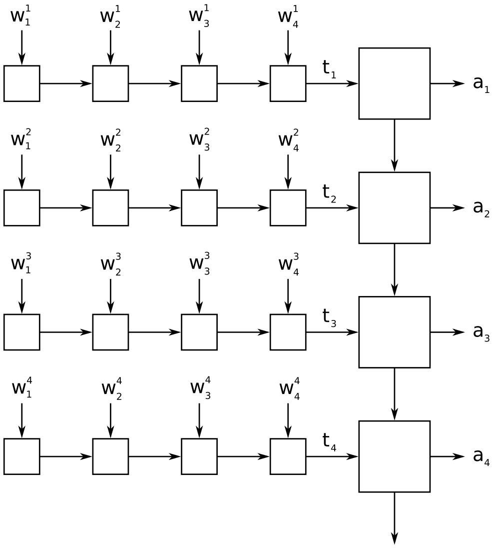


Fig. 1 Two level LSTM architecture used to create embeddings. w_i^j is the word i of turn j, t_j is the learned turn embedding and a_j is the predicted act

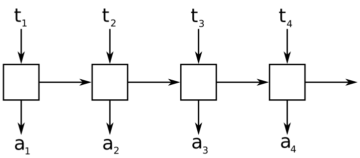


Fig. 2 LSTM architecture used for evaluation. t_i is a fixed pre-trained turn embedding and a_i is the predicted act

For each turn, the corresponding output in the LSTM is used to predict its dialogue act. This architecture is the one used to evaluate the different kinds of fixed pre-trained embeddings that are described in Sect. 4.

4 Skip-Act Vectors

It is possible to construct sentence embeddings using several different means, each of them being able to capture different aspects of a sentence. In our case, we want to find out what kind of embeddings are the best at capturing information about the dialogical structure and the context in which appears a turn. Multiple different kind of embeddings are thus trained on the `DATCHA_RAW` corpus (the large unannotated corpus described in Sect. 5. The following self-supervised embeddings are trained:

Word Average This is simply the average of all the word embeddings in the turn. The word embeddings are learned with FastText [3] on the `DATCHA_RAW` corpus using a dimension of 2048 and a window size of 6. These can be considered as our baseline embeddings since they do not directly take into account the context in which the turns are produced.

Skip-thought These embeddings are learned using a skip-thought model [10]. This model tries to learn the sentence embeddings by trying to regenerate the adjacent sentences during the training. Thus, it tries to learn the context in which a sentence is produced.

In addition to these self-supervised embeddings, we also learned embeddings based on supervised tasks. To learn these embeddings, we use the 2-level LSTM architecture described in Sect. 3. The following supervised embeddings are trained:

RNN Curr Act These embeddings are learned by using a hierarchical neural network that is trained to predict the dialogue act of each turn. The embeddings are the hidden output from the turn layer of the network. Since the `DATCHA_RAW` corpus is not annotated with dialogue acts, we used a system developed during the DATCHA[1] project based on a CRF model developed in [16] (85.7% accuracy) to predict the dialogue acts of each turn of the corpus.

RNN Next Act These embeddings are created similarly to the RNN Curr Act embeddings but instead of predicting the current act for a given turn, the following act is instead predicted.

RNN Prev Act These embeddings are created similarly to the RNN Curr Act embeddings but instead of predicting the current act for a given turn, the previous act is instead predicted.

Skip-Act These embeddings combine the ideas of RNN Prev Act and RNN Next Act by using the same turn layer in the network for both tasks. This model shares the idea of the Skip-thought vectors by trying to learn the context in which the turns are produced. But instead of trying to regenerate the words in the adjacent turns, we try to predict the dialogue acts of the adjacent turns. This allows us to hope that the learned embeddings will focus on the dialogue context of turns. The architecture of this model is presented in Fig. 3.

5 Corpus

Chat conversations are extracted from Orange's customer services contact center logs, and are gathered within the `DATCHA` corpus, with various levels of manual annotations. The `DATCHA` corpus covers a wide variety of topics, ranging from technical issues (e.g.. solving a connection problem) to commercial inquiries (e.g.. purchasing a new offer). They can cover several applicative domains (mobile, internet, tv).

For our experiments, we use two different subsets of these chats:

- Chats from a full month that do not have any gold annotation (79000 dialogues, 3400000 turns) (`DATCHA_RAW`);

[1] http://datcha.lif.univ-mrs.fr.

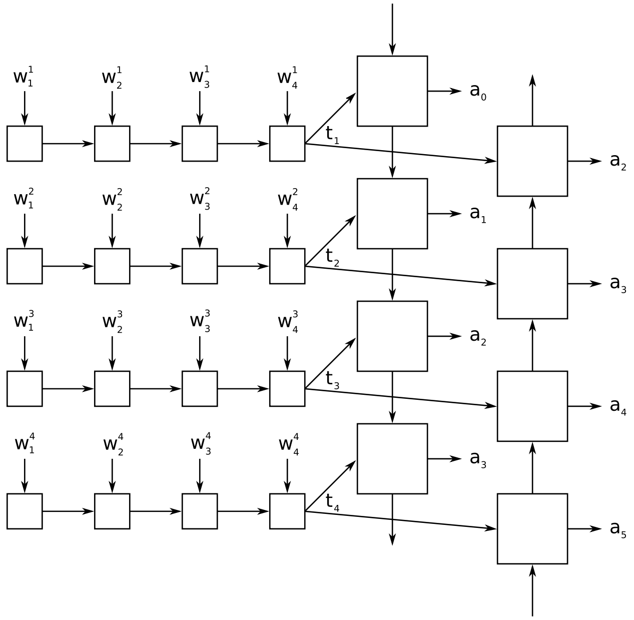


Fig. 3 Architecture used to create skip-act vectors. w_i^j is the word i of turn j, t_j is the learned turn embedding and a_j is the predicted act

- Chats annotated with gold dialogue act annotation (3000 dialogues, 94000 turns) (DATCHA_DA)

These subsets are partitioned into train, test and development parts. The label set used in the dialogue act annotation is as follows:

Label	Meaning	Description
OPE	Opening	Opening turns in the dialogue
PRO	Problem description	The client's description of his problem
INQ	Information question	Turn where a speaker asks for some information
CLQ	Clarification question	A speaker asks for clarification
STA	Statement	New information input
TMP	Temporisation	Starting a break of the dialogue
PPR	Plan proposal	Resolution proposal of the problem
ACK	Acknowledgement	A speaker acknowledges the other speaker's sayings
CLO	Closing	Closing turn
OTH	Other	For turns that don't match other described labels

This set has been designed to be as generic as possible, while taking into account some particular aspects of professional chat interactions (e.g.. *Problem description* or *Plan proposal*). The distribution of the different types of dialogue acts in the test split of the DATCHA_DA corpus can be found in Fig. 4. We also indicate the

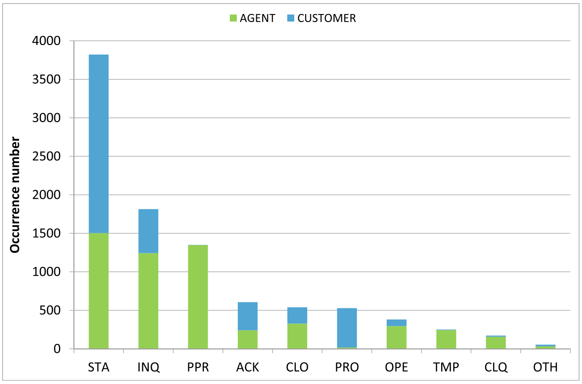


Fig. 4 Dialogue act distribution in the DATCHA_DA test corpus

distributions when considering only a single speaker since they use very different types of turns. For instance, *Plan proposals* are almost exclusively uttered by Agents while, conversely, *Problem descriptions* are mostly observed on Customers side.

6 Turn Embeddings Evaluation

6.1 Evaluation Protocol

We want to make sure that the generated embeddings are able to capture the different aspects of a dialogue. Dialogue acts are one way to partially represent the structure and interactions in a dialogue. Thus, we evaluate the different embeddings on two tasks. For the first task, we try to predict the dialogue act of a turn by only using the sequence of embeddings of the current and previous turns. For the second task, we do the same thing but instead of predicting the dialogue act of the current turn, we predict the act of the next turn (without giving the embedding of the next turn in the input). This second task allows us to tell if the learned embeddings manage to capture information about not only the turn but also about the context in which these turns are produced.

Some of the created embeddings are learned using tasks that involve dialogue acts, thus it is likely that these embeddings obtain the best results. But it is interesting to see if other embeddings are able to obtain similar or close results.

For both tasks, we use the architectures described in Sect. 3 with a hidden size of 512. For each turn, the corresponding output in the RNN is given to a decision layer which uses a softmax to output a probability distribution of the dialogue acts. We use cross-entropy as our loss function and Adam as the optimizer with a learning rate of 0.001. The PyTorch framework is used to build the different architectures.

In order to evaluate the quality of the different predictions, we primarily use 2 metrics:

- **accuracy**: the percentage of correct decisions;
- **macro F1**: the non-weighted average of the F1-measures of the 10 act labels. The F1-measure is the harmonic mean of precision P and recall R for a given label l such as $F1(l) = \frac{2 \times P(l) \times R(l)}{P(l)+R(l)}$;

6.2 Results and Analyses

Results of the prediction of the current and next acts are reported in Table 1. The first line corresponds to the first model described in Fig. 1 where no pre-trained embeddings are used and where the embeddings are learned jointly with the model's parameters on the DATCHA_DA corpus. The following lines correspond to the single turn-level architecture presented in Fig. 2 using several variants of fixed turn embeddings, pre-trained on the large DATCHA_RAW corpus. For each embedding type and task, we only report the results of the configuration that obtained the best results. We can first note a big difference in performances between the two tasks with the next act task being much harder than the current act task. It seems to be very difficult to predict the next act given the history of turns, particularly for some of them, as can be seen in Figs. 5 and 6 where some acts such as CLQ, INQ or PPR see a drop of 60 points in their F1-score while acts such as STA, CLO or OPE only have a drop of 20 points. This could be explained by the fact that closings and openings are easier to locate in the conversation, while statements are the most represented labels in conversations. On the other hand, it is not necessarily easy to know that the next turn is going to be a question or a plan proposal. We can also notice that the OTH act is not at all correctly predicted in the next act task, and even in the current act task it is the label with the worst F1-score. This is probably due to the fact that turns that are labeled OTH are usually filled with random symbols or words and are both very diverse and not frequent.

Table 1 Evaluation of the prediction of the current and next dialogue acts on all turns

		Current act		Next act	
LSTM architecture	Pre-trained embeddings	Accuracy	Macro-F1	Accuracy	Macro-F1
2-level hierarchical	None	83.69	78.15	46.21	26.45
Turn level	Word average	82.96	79.47	48.26	30.09
Turn level	Skip-thought	82.50	75.73	48.30	28.61
Turn level	RNN curr act	84.74	80.47	48.54	31.42
Turn level	RNN next act	84.40	81.42	49.97	34.47
Turn level	RNN prev act	83.02	80.44	48.77	31.96
Turn level	**Skip-act**	**85.24**	**82.16**	**49.96**	**35.33**

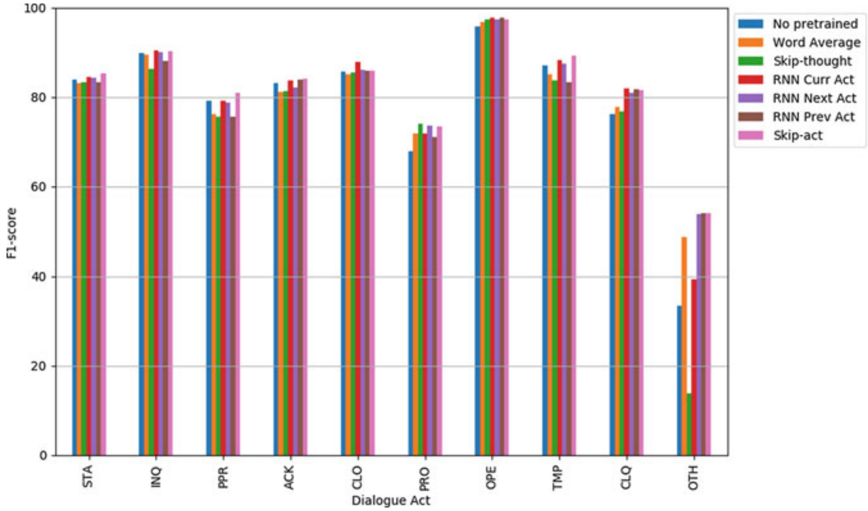


Fig. 5 F1-scores on the current act task on all turns

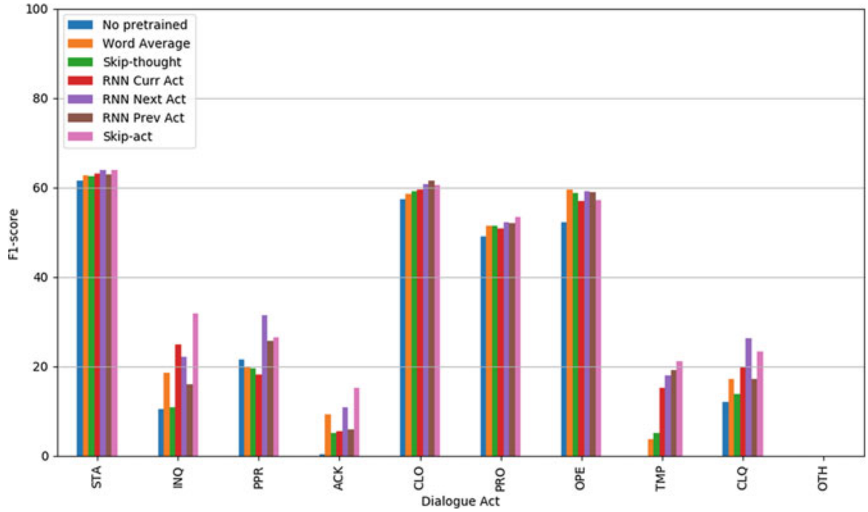


Fig. 6 F1-scores on the next act task on all turns

Unsurprisingly, for both tasks, the best results are obtained with embeddings learned using dialogue acts. However, the **Word Average** and **Skip-thought** vectors both achieve good results but they still are 2 points lower than the best results. It is interesting to note that the **Skip-thought** vectors do not achieve better results than **Word Average** vectors on the next act task. This can be surprising since they would have been expected to better capture information about the surrounding turns, however the generalization from word level prediction to turn level prediction is not sufficiently efficient. It is also interesting to note that better results are achieved by **RNN Curr Act** embeddings (84.74%), which are learned on a corpus with a noisy annotation, compared to results achieved by the embeddings learned during the training on the DATCHA_DA corpus (83.69%) which has gold annotation. This results confirms our choice to train turn embeddings separately with light supervision on a significantly larger, even though noisy, training corpus.

Another interesting aspect of these results is the comparison of the different kinds of embeddings learned with dialogue act related tasks. Indeed, on the current act task, we can notice that **RNN Curr Act** embeddings obtain slightly lower results (−0.5 points) than **Skip-act** embeddings. This is surprising since **RNN Curr Act** are learned using the same task than the evaluation, while **Skip-act** are learned by trying to predict the next and previous acts only. These results could mean that **Skip-act** are more robust since they learn in what context the acts are produced. On the next act task, both the **RNN Next Act** and **Skip-act** achieve the same performances with 50% accuracy, while the **RNN Curr Act** embeddings obtain an accuracy of 48.5%.

We also reported in Tables 2 and 3 the results when considering only the turns from respectively the agent and the client for evaluation. It is important to note that the label distribution is very different depending on the speaker. Most of the questions (CLQ and INQ) and nearly all plan proposals (PPR) and temporisations (TMP) are from the agent while most of the problem descriptions (PRO) and the majority of statements (STA) are from the client. When evaluated on the agent side, **Skip-act** embeddings are again the best embeddings for both tasks, being 1 point higher than the **RNN Next Act** embeddings and 3.5 points higher than the **RNN Curr Act** embeddings. These results are interesting since the agent is the speaker with the most variety in the types of turns, including many turns with questions, plan proposals or temporisations. This seems to indicate that **Skip-acts** manage to capture more information about the dialogue context than the other embeddings.

Table 2 Evaluation of the prediction of the current and next dialogue acts on agent's turns

		Current act		Next act	
LSTM architecture	Pre-trained embeddings	Accuracy	Macro-F1	Accuracy	Macro-F1
2-level hierarchical	None	84.22	77.38	35.87	23.16
Turn level	Word average	82.48	77.31	37.78	27.02
Turn level	Skip-thought	80.36	74.75	37.07	25.39
Turn level	RNN curr act	84.70	79.01	38.90	29.00
Turn level	RNN next act	84.30	82.42	41.29	32.60
Turn level	RNN prev act	83.24	80.11	38.80	28.81
Turn level	**Skip-act**	**85.48**	**82.94**	**42.30**	**33.56**

Table 3 Evaluation of the prediction of the current and next dialogue acts on customer's turns

		Current act		Next act	
LSTM architecture	Pre-trained embeddings	Accuracy	Macro-F1	Accuracy	Macro-F1
2-level hierarchical	None	83.01	58.58	59.48	21.13
Turn level	Word average	83.59	60.97	61.71	21.80
Turn level	Skip-thought	85.31	59.13	62.70	20.49
Turn level	RNN curr act	84.78	64.16	60.89	21.74
Turn level	RNN next act	84.54	63.20	61.09	22.91
Turn level	RNN prev act	82.74	61.88	61.56	21.73
Turn level	**Skip-act**	**84.93**	**63.99**	**59.78**	**23.79**

We can also notice that this time, **Skip-thought** vectors obtain lower results than the simple **Word Average**. When evaluated on the customer side, **Skip-thought** vectors obtain the best scores on both tasks when looking at the accuracy (85.31% and 62.70%) but lower scores in terms of macro-F1. The scores on the next act task are higher but this is only due to the fact that the `STA` act represents 57.4% of the samples, whereas on all the turns and for the agent they respectively represent 40.2% and 27.8% of the samples.

7 Conclusion

We have proposed a new architecture to compute dialogue turn embeddings. Within the skip-act framework, a multitask model is trained in order to jointly predict the previous and the next dialogue acts. Trained in a lightly supervised way on a large corpus of chat conversations with an automatic dialogue act annotation, the output of the common hidden layer provides an efficient turn level vector representation that tends to capture the dialogic structure of the interactions. We have evaluated several dialogue turn embeddings configurations on two tasks, first predicting the associated dialogue act of the current turn, and then predicting the next dialogue act which is a more challenging task requiring a better representation of the dialogue structure. Skip-act embeddings achieve the best results on both tasks. In the future, it would be interesting to combine skip-thoughts and skip-acts in order to be able to capture the semantic and syntactic information in addition to the dialogue context of turns.

Acknowledgements This work has been partially funded by the Agence Nationale pour la Recherche (ANR) through the following programs: ANR-15-CE23-0003 (DATCHA), ANR-16-CONV-0002 (ILCB) and ANR-11-IDEX-0001-02 (A*MIDEX).

References

1. Agirre E, Banea C, Cardie C, Cer D, Diab M, Gonzalez-Agirre A, Guo W, Mihalcea R, Rigau G, Wiebe J (2014) Semeval-2014 task 10: multilingual semantic textual similarity. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pp 81–91
2. Auguste J, Rey A, Favre B (2017) Evaluation of word embeddings against cognitive processes: Primed reaction times in lexical decision and naming tasks. In: Proceedings of the 2nd workshop on evaluating vector space representations for NLP, pp 21–26. Copenhagen, Denmark
3. Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans Assoc Comput Linguist 5(1):135–146
4. Bruni E, Boleda G, Baroni M, Tran NK (2012) Distributional semantics in technicolor. In: Proceedings of the 50th annual meeting of the association for computational linguistics: long papers, vol 1, pp 136–145. Association for Computational Linguistics, Stroudsburg
5. Conneau A, Kiela D (2018) SentEval: an evaluation toolkit for universal sentence representations. In: Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018). Miyazaki, Japan
6. Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A (2017) Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of the 2017 conference on empirical methods in natural language processing, pp 670–680 (2017)
7. Conneau A, Kruszewski G, Lample G, Barrault L, Baroni M (2018) What you can cram into a single \$&!#* vector: probing sentence embeddings for linguistic properties. In: Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long Papers), pp 2126–2136. Association for Computational Linguistics, Melbourne, Australia
8. Hill F, Cho K, Korhonen A (2016) Learning distributed representations of sentences from unlabelled data. In: Proceedings of NAACL-HLT, pp 1367–1377
9. Hill F, Reichart R, Korhonen A (2016) Simlex-999: Evaluating semantic models with (genuine) similarity estimation. Computational Linguistics 00173
10. Kiros R, Zhu Y, Salakhutdinov RR, Zemel R, Urtasun R, Torralba A, Fidler S (2015) Skip-thought vectors. In: Advances in neural information processing systems, pp 3294–3302
11. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning, pp 1188–1196
12. Luong T, Socher R, Manning CD (2013) Better word representations with recursive neural networks for morphology. In: CoNLL, pp 104–113. 00192
13. Marelli M, Menini S, Baroni M, Bentivogli L, Bernardi R, Zamparelli R (2014) A SICK cure for the evaluation of compositional distributional semantic models. In: LREC, pp 216–223
14. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. In Proceedings of workshop at ICLR. 03267
15. Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: EMNLP, vol 14, pp 1532–1543. 01307
16. Perrotin R, Nasr A, Auguste J (2018) Dialog acts annotations for online chats. In: 25e Conférence Sur Le Traitement Automatique Des Langues Naturelles (TALN). Rennes, France
17. Pragst L, Rach N, Minker W, Ultes S (2018) On the vector representation of utterances in dialogue context. In: LREC
18. Søgaard A (2016) Evaluating word embeddings with fMRI and eye-tracking. ACL 2016, p 116. 00000

End-to-end Modeling for Selection of Utterance Constructional Units via System Internal States

Koki Tanaka, Koji Inoue, Shizuka Nakamura, Katsuya Takanashi, and Tatsuya Kawahara

Abstract In order to make conversational agents or robots conduct human-like behaviors, it is important to design a model of the system internal states. In this paper, we address a model of favorable impression to the dialogue partner. The favorable impression is modeled to change according to user's dialogue behaviors and also affect following dialogue behaviors of the system, specifically selection of utterance constructional units. For this modeling, we propose a hierarchical structure of logistic regression models. First, from the user's dialogue behaviors, the model estimates the level of user's favorable impression to the system and also the level of the user's interest in the current topic. Then, based on the above results, the model predicts the system's favorable impression to the user. Finally, the model determines selection of utterance constructional units in the next system turn. We train each of the logistic regression models individually with a small amount of annotated data of favorable impression. Afterward, the entire multi-layer network is fine-tuned with a larger amount of dialogue behavior data. An experimental result shows that the proposed method achieves higher accuracy on the selection of the utterance constructional units, compared with methods that do not take into account the system internal states.

K. Tanaka · K. Inoue (✉) · S. Nakamura · K. Takanashi · T. Kawahara
Graduate School of Informatics, Kyoto University, Kyoto, Japan
e-mail: inoue@sap.ist.i.kyoto-u.ac.jp

K. Tanaka
e-mail: tanaka@sap.ist.i.kyoto-u.ac.jp

S. Nakamura
e-mail: shizuka@sap.ist.i.kyoto-u.ac.jp

K. Takanashi
e-mail: takanashi@sap.ist.i.kyoto-u.ac.jp

T. Kawahara
e-mail: kawahara@sap.ist.i.kyoto-u.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_2

1 Introduction

It is important for spoken dialogue systems to introduce internal states in order to realize human-like dialogue. By taking into account both input user utterances and system internal states, spoken dialogue systems are expected to generate more human-like natural utterances. Emotion has been considered as an internal state for spoken dialogue systems and virtual agents [2, 3, 13].

We address *favorable impression* to a user as an internal state of the system. We set up a speed-dating dialogue task where a male user talks with a female conversational robot about their profiles. In human-human speed-dating dialogue, their behaviors and attitudes sometimes reflect the degree of favorable impression to their interlocutors [9, 12]. In this study, to express the degree of favorable impression, we propose a dialogue system that selects utterance constructional units, inspired by a series of studies on the discourse analysis [17]. The utterance constructional units contain three parts: *response*, *episode*, and *question*. *Response* is a reaction to the user's utterance, such as feedbacks and answers to questions. *Episode* corresponds to information given by the system such as self-disclosure. *Question* is made by the system toward the user to elaborate the current topic or change the topic. Figure 1 illustrates the main idea of our proposed system. For example, when the degree of favorable impression to the user is high, the system tends to select multiple units such as the combination of *response* and *episode*, or another combination of *response* and *question*, to be more talkative. On the other hand, when the degree is low, the system would select only *response*.

We realize selection of utterance constructional units by a hierarchical structure of logistic regression models. The input is a set of features based on the user's dialogue behaviors. The output is a selection of the utterance constructional units of the next system turn. In the intermediate layer of the hierarchical structure, the degree of favorable impression is represented as an internal state. The proposed model predicts the favorable impression to the user and then the utterance constructional units step by step, where each step is realized with a logistic regression model. We train each logistic regression model with annotated labels of the favorable impression to the user. However, it is difficult to obtain a large number of training labels for the internal states. On the other hand, it is easier to get a large amount of data for the input and output behaviors because these are actual behaviors that can be objectively defined

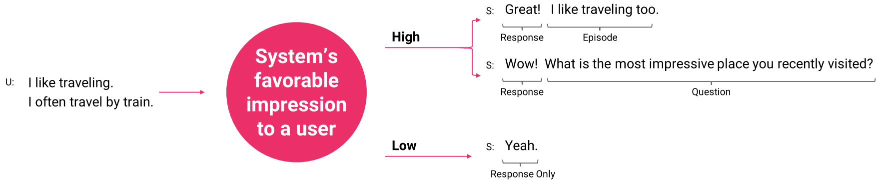


Fig. 1 Main idea of the proposed system that selects the next system utterance based on the system's favorable impression toward the user (U: user, S: system)

and observed in dialogue corpora. In this paper, we also propose an efficient model training to leverage the benefits of making use of internal states. At first, we pre-train each logistic regression model with a small number of training labels of the internal states. We then fine-tune the whole neural network with a larger amount of data of the input and output behaviors in an end-to-end manner. The pre-training captures the internal states, and the end-to-end fine-tuning scales up the amount of training data, which is vital for robust training. This study contributes to realizing dialogue systems that model internal states and also efficient model training where the amount of training data for the internal states is limited.

2 Speed-Dating Human-Robot Dialogue Corpus

In this section, we explain the dialogue data used in this study. We recorded a set of speed-dating dialogues where a male human subject talked with a female humanoid robot that was operated by another female subject. Right after the recording, we took a survey to obtain training labels of the internal states. We also manually annotated the utterance constructional units on the recorded dialogue data.

2.1 Dialogue Data Collection

We have collected a series of speed-dating dialogues between a male subject and a female humanoid robot named ERICA [7, 10]. ERICA was operated by another human subject, called an operator, who was in a remote room. When the operator spoke, the voice was directly played with a speaker placed on ERICA, and the lip and head motion of ERICA was automatically generated [8, 14]. The operator also controlled ERICA's behaviors such as eye-gaze, head nodding, and arm gestures. The snapshot of this data collection is shown in Fig. 2. We recorded 18 dialogue sessions which lasted 10 min and 55 s on average. The human subjects were 18 male university

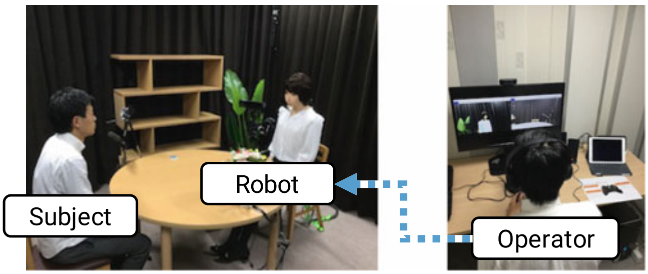


Fig. 2 Snapshot of data collection in WoZ setting

students (both undergraduate and graduated students). The ERICA's operators were 4 actresses whose ages ranged from 20s to 30s. Whereas each human subject participated in only one dialogue session, each ERICA's operator participated in several sessions. They are all native Japanese speakers. We used multimodal sensors that consisted of microphones, a microphone array, RGB cameras, and Kinect v2. We manually annotated utterances, backchannels, laughing, fillers, dialogue turns, and dialogue acts using recommended standards [5].

The dialogue scenarios and instructions are as follows. Since they met each other for the first time, they had to exchange their personal information to know well each other. In advance, we gave the participants a list of conversational topics that are likely to be talked about in first-encounter dialogues, such as hobbies, occupation, and hometown. We then instructed the participants to make a conversation based on the topic list. In the actual dialogue, participants often talked about the topics on the list such as favorite movies, sports, food, and recent trips. For the ERICA's operator, we instructed how to select the utterance constructional units together with the concept of the favorable impression. We asked the operator to select the utterance constructional units based on the degree of her favorable impression to the subject, but we also told that she did not necessarily need to follow this to keep the dialogue natural. We also told that the operator did not need to entertain the subject and the degree of her favorable impression to the subject could be not only positive but also negative.

After each dialogue session, we asked the operator to answer a survey. After the operator listed dialogue topics that they talked about, she rated the following items for each topic on the 7-point scale.

1. Operator's favorable impression to the subject
2. Subject's favorable impression to ERICA estimated by the operator
3. Operator's interest in the topic
4. Subject's interest in the topic estimated by the operator

The favorable impression is represented in one-dimension, positive and negative, as we regard it as a specific indicator in first-encounter dialogue. Although we conducted a similar survey to the male subjects, we used only the survey result from the operators. The reason is that the male subject was a different person on each dialogue session while the operators' survey should be consistent among sessions.

2.2 Analysis

First, we segmented all utterances by dialogue turns. In total, the number of turns of the operators was 899. Then, we manually annotated a set of utterance constructional units for each turn. This annotation was made by one annotator. The distribution of the patterns of utterance constructional units is reported in Table 1. As we see from the table, the majority of the patterns of utterance constructional units was response only (472 samples). Notably, the operators occasionally gave their episode and asked

Table 1 Distribution of the pattern of utterance constructional units

Utterance constructional units			Frequency
Response	*Episode*	*Question*	
✓	–	–	472
✓	✓	–	177
✓	–	✓	86
–	✓	–	69
–	–	✓	53
✓	✓	✓	8
Others			34
Total			899

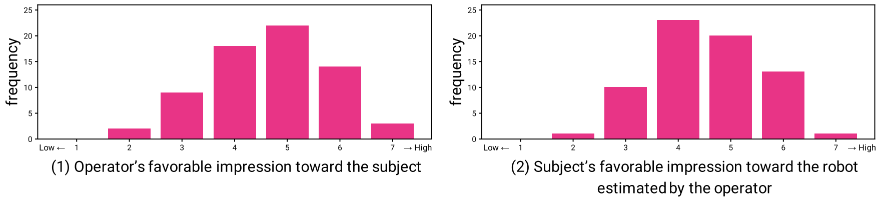


Fig. 3 Distribution of favorable impression reported by ERICA's operators

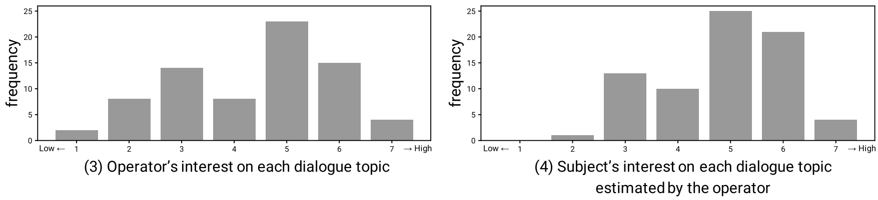


Fig. 4 Distribution of interest reported by ERICA's operators

back questions, but the cases having both an episode and a question was very rare (8 samples). We hypothesize that the operators reflected their favorable impression to the subjects on the utterance constructional units.

We analyzed the survey results from the operators on the following items: (1) operator's favorable impression to the user, (2) subject's favorable impression to ERICA estimated by the operator, (3) operator's interest in each dialogue topic, and (4) subject's interest in each dialogue topic estimated by the operator. The distributions of the four items are plotted in Figs. 3 and 4. The number of dialogue topics was 74 in total. The distributions of interest tended to be more varied than those of favorable impression. This result suggests that the degree of interest more depends on the dialogue topics. On the other hand, this result also suggests that the favorable impression is more stable and gradually changes during the dialogue.

3 Problem Formulation

The task of this study is to select the utterance constructional units of the next system turn based on observed behavior features of the user. The problem formulation is illustrated in Fig. 5. The input feature vector is based on both the speaking and listening behaviors of the user. The speaking behavior feature is extracted during the preceding user turn, referred as $\boldsymbol{o}_s$. The listening behavior feature is computed during the last system turn, referred as $\boldsymbol{o}_l$. We concatenate the behavior feature vectors as:

$$\boldsymbol{o} := (\boldsymbol{o}_s \ , \ \boldsymbol{o}_l) \ . \tag{1}$$

The detail of the feature set is explained in Sect. 5. The output is the pattern of the utterance constructional units that consists of three elements: *response*, *episode*, and *question*. We refer the output as a system action $\boldsymbol{a}$. In this study, we take into account the internal states such as the system's favorable impression to the user. We define the internal states as a vector $\boldsymbol{s}$. In summary, the problem in this study is to predict the next system action $\boldsymbol{a}$ from the observation behaviors $\boldsymbol{o}$ by considering the internal states $\boldsymbol{s}$. This is a typical formulation in conventional studies on spoken dialogue systems where the internal states $\boldsymbol{s}$ correspond to dialogue states of slot filling. In the case of conventional studies such as task-oriented dialogues, the dialogue states were defined clearly and objectively, which makes it easy to collect a large number of training labels for statistical dialogue models such as Markov decision process (MDP) and partially observable Markov decision process (POMDP) [20]. In the current study on the first-encounter dialogue, however, the internal states correspond to states such as favorable impression. These states are ambiguous and subjective,

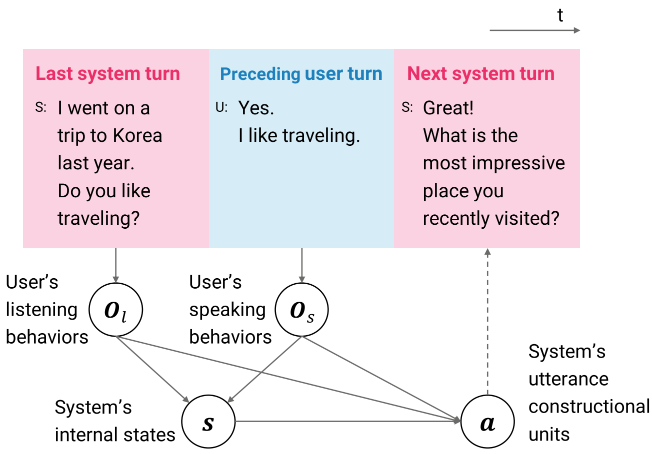


Fig. 5 Problem formulation for considering internal states to select the system next action

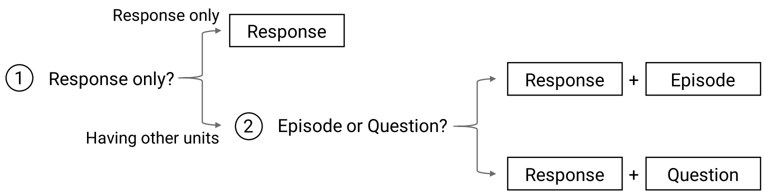


Fig. 6 Taxonomy for selection of the utterance constructional units. The numbers (1 and 2) in the figure correspond to classification tasks

which makes it difficult to prepare a sufficient number of training labels of them. Therefore, we propose efficient end-to-end training by facilitating a small number of labels of the internal states.

Since the distribution of the utterance constructional units is skewed as shown in Table 1, we do not directly select the utterance constructional units. Instead, we divide this problem into the following two sub-tasks. These sub-tasks can be defined as a taxonomy depicted in Fig. 6. The first task is to decide whether the system's turn consists of a response only or have other units (an episode and/or a question). If the decision is the latter case, the system triggers the second task which is to decide whether the system generates an episode or a question. Since we could observe only a few samples where all three utterance constructional units were used at the same time, we do not consider this rare case in the current formulation. In this study, we make the selection model for each task independently, but we combine them to decide the pattern of the utterance constructional units finally. The distribution and definition of labels of the utterance constructional units are summarized in Table 2. The first task corresponds to the selection between the majority pattern and the others. The second task focuses on the remainder steps.

4 End-to-end Modeling Using a Small Number of Labels of Internal States

We take into account the internal states such as favorable impression to the user in order to select the utterance constructional units of the next system turn. However, the number of training labels of the internal states is limited. Actually, in the current study, we could obtain the labels of favorable impression and interest only on each topic, whereas we have to generate the system's action for every turn. This is a universal problem in modeling internal states. On the other hand, we can easily obtain the labels of behaviors such as the observation $\boldsymbol{o}$ and the action $\boldsymbol{a}$ because these behaviors can be objectively observed.

We propose efficient end-to-end modeling for the selection of the utterance constructional units by using a small number of labels of the favorable impression and the interest. The proposed model is based on hierarchical neural networks where

Table 2 Definition of labels of the utterance constructional units for each task (p: positive sample, n: negative sample, –: not used)

Utterance constructional units				Task	
Response	*Episode*	*Question*	Freq.	1	2
✓	–	–	472	p	–
✓	✓	–	177	n	p
✓	–	✓	86	n	n
–	✓	–	69	n	p
–	–	✓	53	n	n
✓	✓	✓	8	n	–
Others			34		
Total			899		

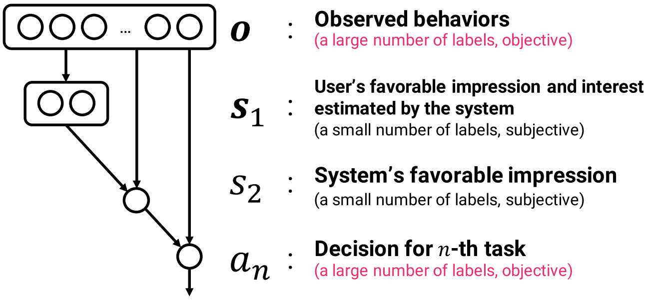


Fig. 7 Proposed model considering internal states as hidden layers of the network

the internal states are represented as hidden layers. Figure 7 depicts an overview of the proposed model. First, we train each layer one by one. For example, we train a prediction model for the user's favorable impression to the system based on the observed behaviors of the user ($\boldsymbol{o}$). This pre-training is done with a small number of labels of the internal states. After we train each layer, the entire network is fine-tuned with a much larger number of data sets of the observation $\boldsymbol{o}$ and the output system action $\boldsymbol{a}$.

4.1 Network Architecture

The proposed hierarchical neural network estimates the internal states step by step. The network architecture is depicted in Fig. 7. We observe the input feature vector $\boldsymbol{o}$. The dimension of the input vector is D_o. Although it is possible to directly predict the system's favorable impression to the user from the observation, we first estimate the user's favorable impression to the system and the interest on the current topic as:

$$\boldsymbol{s}_1 = \sigma(A_1 \boldsymbol{o}^T + \boldsymbol{b}_1^T) , \tag{2}$$

where $\boldsymbol{s}_1$ is a two-dimension vector corresponding to the values of the user's favorable impression and interest estimated by the system. A_1 and $\boldsymbol{b}_1$ are network parameters whose sizes are $2 \times D_o$ and 2, respectively. $\sigma()$ is the sigmoid function and T represents the transpose. Next, we predict the system's favorable impression to the user from both the user's favorable impression to the system and the interest estimated in the previous step and also from the observation (referred as $\boldsymbol{s}_{1'} = (\boldsymbol{s}_1, \boldsymbol{o})$):

$$s_2 = \sigma(A_2 \boldsymbol{s}_{1'}^T + b_2) , \tag{3}$$

where s_2 is a scalar corresponding to the value of the system's favorable impression to the user. A_2 and b_2 are network parameters whose sizes are $1 \times (2 + D_o)$ and 1, respectively. Finally, we calculate the probability for each task of the selection of the utterance constructional units in the same manner as:

$$a_n = \sigma(A_3 \boldsymbol{s}_{2'}^T + b_3) , \tag{4}$$

where $\boldsymbol{s}_{2'}$ is a concatenated vector consisting of the predicted system's favorable impression to the user and the observation as $\boldsymbol{s}_{2'} = (s_2, \boldsymbol{o})$, and a_n is the probability for the n-th task which was defined as a binary classification defined in Sect. 3. A_3 and b_3 are network parameters whose sizes are $1 \times (1 + D_o)$ and 1, respectively. In this study, we solve the two tasks individually. We train the above model for each task, and the set of the output scalar values make a final system action $\boldsymbol{a}$.

4.2 Model Training

The model training consists of two steps: pre-training and fine-tuning. First, we train each layer step by step as pre-training. Since we have labels of the internal states on each topic, we assume the internal states are unchanged during the same topic. This limitation also means that it is difficult to scale up the number of labels of the internal states. Therefore, we fine-tune the entire network with a larger number of labels of the observation $\boldsymbol{o}$ and the system action $\boldsymbol{a}$ through back-propagation. To keep the effect of the pre-training, we add the square error between the model parameters by the pre-training and those after the fine-tuning to the loss function as:

$$E'(W) = E(W) + SE(W, W_{pre}) , \tag{5}$$

where $E(W)$ is the loss function of the output layer of the network, and $SE(W, W_{pre})$ is the square error between the model parameters by the pre-training (W_{pre}) and those after the fine-tuning (W). Specifically, we summed squared Frobenius norm of the difference of each model parameter to calculate $SE(W, W_{pre})$.

5 Feature Set

In order to implement the proposed system, we need to define the observation vector $\boldsymbol{o} = (\boldsymbol{o}_s, \boldsymbol{o}_l)$. We use both features of speaking and listening behaviors of the user. The features are chosen based on previous studies on emotion and interest recognition [1, 15, 16, 18, 19]. We manually annotated the following features and used them in the experiment, although the majority of these features can be computed automatically.

The features of speaking behaviors $\boldsymbol{o}_s$ are calculated from the preceding user turn and listed below.

- Turn duration
- Pause duration between the end of the last system turn and the beginning of the preceding user turn
- Voice activity ratio
- Global voice activity ratio from the beginning of the dialogue until the end of the preceding user turn
- Speech rate
- Intensity (mean, range)
- F0 (mean, range)
- Length of *episode* (if the turn contains *episode* otherwise zero)
- Laughter frequency
- Filler frequency (short phrases that fill a pause within a turn, such as "*uh*")
- Pattern of utterance constructional units

We used the Praat [4] software to extract intensity and F0 from the user utterances. We approximated the length of *episode* as the number of long utterance units (LUUs) [6]. The LUUs are defined to approximate semantic units so that we intended to capture the substantial volume of the episode. The pattern of the utterance constructional units of the user turn is represented as binary vectors where each dimension corresponds to the occurrence of each element of the utterance construction unit. The dimension of the vector $\boldsymbol{o}_s$ is 18.

The features of listening behaviors $\boldsymbol{o}_l$ are calculated from the last system turn and listed below.

- Backchannel frequency (such as "*yeah*")
- Laughter frequency

The dimension of the vector $\boldsymbol{o}_l$ is 2. We squeezed the feature set to these because this is the first step of the study. In future work, we will consider the use of additional listening behaviors such as eye gaze and head nodding.

6 Experimental Evaluation

We evaluated the proposed method with the first-encounter dialogue corpus described in Sect. 2. Five-fold cross validation was conducted to calculate an average precision, recall, and F1 score. We implemented the neural network model with TensorFlow 1.7.0. We used Adam [11] as the optimization method and empirically set the learning rate at 10^{-2} for the first task and 10^{-6} for the second task. We prepared three compared models. The first model is to directly predict the utterance constructional units from the observation with a one-layer neural network, which is equivalent to a logistic regression model, referred as *baseline*. The second model has the same architecture as the proposed model in that it is a multi-layer neural network, but the pre-training is not conducted. Instead, the network parameters are initialized with random values, referred as *w/o. pre-training*. The third model is same as the proposed model, but the fine-tuning is not conducted, referred as *w/o. fine-tuning*.

As shown in Fig. 6, we solve two different tasks for the selection of the utterance constructional units: (1) response only or having other units, (2) generate an episode or a question. The ratios of positive samples (chance levels) in the whole data set are 0.527 and 0.605 on the first and second tasks, respectively. The results of the two prediction tasks are reported in Tables 3 and 4. Overall, the proposed method outperformed the *baseline* model and the *w/o. pre-training* model in both tasks. This shows that modeling and pre-training the internal states is effective in the proposed model. Furthermore, the combination of the pre-training and the fine-tuning improves the model performance. The fine-tuning makes it possible to train with a larger number of labels, which is an advantage of the use of hierarchical neural networks.

Table 3 Prediction result on the first task (response only or having other units)

Model	Precision	Recall	F1
Baseline	0.667	0.658	0.662
w/o. pre-training	0.628	0.643	0.635
w/o. fine-tuning	0.708	0.586	0.641
Proposed	0.679	0.674	0.677

Table 4 Prediction result on the second task (episode or question)

Model	Precision	Recall	F1
Baseline	0.617	0.748	0.676
w/o. pre-training	0.649	0.740	0.692
w/o. fine-tuning	0.664	0.784	0.719
Proposed	0.666	0.788	0.722

7 Conclusions

We have proposed a model that selects the utterance constructional units from the observed user behaviors by taking into account internal states such as favorable impression to interlocutors. The utterance constructional units were defined as the combination of three components: *response*, *episode*, and *question*. The proposed model is a hierarchical neural network that represents the internal states as hidden layers. The number of training labels of the internal states is limited so that we pre-trained each layer with a small number of labels one layer by one layer. Afterward, we fine-tuned the whole network with a larger number of training data of behaviors that can be objectively measured. This approach will be useful for systems with internal states that can have a small number of training data. We evaluated the system with the speed-dating dialogue corpus, and showed the proposed model achieved better prediction performance than the compared methods that did not take into account the system internal states. Although we dealt with the task for the three utterance constructional units, the proposed approach is not limited to this task.

In future work, we will implement the proposed system in a live spoken dialogue system to evaluate in real applications. It is needed to implement response generation using the prediction result of the utterance constructional units. Additionally, we plan to use multi-modal behaviors such as eye-gaze and head nodding.

Acknowledgements This work was supported by JST ERATO Grant Number JPMJER1401, Japan. The authors would like to thank Professor Graham Wilcock for his insightful advice.

References

1. Anagnostopoulos CN, Iliou T, Giannoukos I (2015) Features and classifiers for emotion recognition from speech: a survey from 2000 to 2011. Artif Intell Rev 43(2):155–177
2. Bates J (1994) The role of emotion in believable agents. Commun ACM 37(7):122–125
3. Becker C, Kopp S, Wachsmuth I (2004) Simulating the emotion dynamics of a multimodal conversational agent. In: ADS, pp. 154–165
4. Boersma P (2001) Praat, a system for doing phonetics by computer. Glot Int. 5(9):341–345
5. Bunt H, Alexandersson J, Carletta J, Choe JW, Fang AC, Hasida K, Lee K, Petukhova V, Popescu-Belis A, Romary L, et al (2010) Towards an ISO standard for dialogue act annotation. In: LREC, pp. 2548–2555
6. Den Y, Koiso H, Maruyama T, Maekawa K, Takanashi K, Enomoto M, Yoshida N (2010) Two-level annotation of utterance-units in japanese dialogs: an empirically emerged scheme. In: LREC, pp. 1483–1486
7. Inoue K, Milhorat P, Lala D, Zhao T, Kawahara T (2016) Talking with erica, an autonomous android. In: SIGDIAL, pp 212–215
8. Ishi CT, Ishiguro H, Hagita N (2012) Evaluation of formant-based lip motion generation in tele-operated humanoid robots. In: IROS, pp 2377–2382
9. Jurafsky D, Ranganath R, McFarland D (2009) Extracting social meaning: identifying interactional style in spoken conversation. In: NAACL, pp 638–646
10. Kawahara T (2018) Spoken dialogue system for a human-like conversational robot ERICA. In: IWSDS

11. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: ICLR
12. Pentland AS (2010) Honest signals: how they shape our world. MIT press, Cambridge (2010)
13. Picard RW (1997) Affective computing, vol 252. MIT press, Cambridge
14. Sakai K, Ishi CT, Minato T, Ishiguro H (2015) Online speech-driven head motion generating system and evaluation on a tele-operated robot. In: ROMAN, pp 529–534
15. Schuller B, Köhler N, Müller R, Rigoll G (2006) Recognition of interest in human conversational speech. In: ICSLP, pp 793–796
16. Schuller B, Steidl S, Batliner A, Vinciarelli A, Scherer K, Ringeval F, Chetouani M, Weninger F, Eyben F, Marchi E, et al (2013) The INTERSPEECH 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. In: Interspeech, pp 148–152
17. Sinclair JM, Coulthard M (1975) Towards an analysis of discourse: the English used by teachers and pupils. Oxford University Press, Oxford
18. Wang WY, Biadsy F, Rosenberg A, Hirschberg J (2013) Automatic detection of speaker state: lexical, prosodic, and phonetic approaches to level-of-interest and intoxication classification. Comput Speech Lang 27(1):168–189
19. Wu CH, Lin JC, Wei WL (2014) Survey on audiovisual emotion recognition: databases, features, and data fusion strategies. APSIPA Trans Signal Inf Process 3:1–18
20. Young S, Gašić M, Thomson B, Williams JD (2013) Pomdp-based statistical spoken dialog systems: a review. Proc IEEE 101(5):1160–1179

Context Aware Dialog Management with Unsupervised Ranking

Svetlana Stoyanchev and Badrinath Jayakumar

Abstract We propose *MoveRank*, a novel hybrid approach to dialog management that uses a knowledge graph domain structure designed by a domain-expert. The domain encoder converts a symbolic output of the NLU into a vector representation. *MoveRank* uses an unsupervised similarity measure to obtain the optimal dialog state modifications in a given context. Using a 1K utterance dataset automatically constructed with template expansion from a small set of annotated human-human dialogs, we show that the proposed unsupervised ranking approach produces the correct result on the *gold labeled* input without spelling variations. Using an encoding method designed to handle spelling variations, *MoveRank* is correct with $F - 1 = 0.86$, with the complete set of labels (including *intent, entity,* and *item*) and $F - 1 = 0.82$, with only the *intent* labels.

1 Introduction

In this paper, we describe *MoveRank*, a novel approach to dialog management. *MoveRank* applies to the knowledge-graph driven framework (KGD) described in [15]. With the KGD framework, a domain-specific dialog manager is authored declaratively in the form of a knowledge graph. KGD extends the idea of system generation from a form [16] to handle multiple tasks/forms. The proposed approach addresses error propagation from the speech recognition (ASR) and natural language understanding (NLU) output using context and domain structure. *MoveRank* is a more

S. Stoyanchev (✉)
Toshiba Europe, 208 Cambridge Science Park, Cambridge, UK
e-mail: svetlana.stoyanchev@crl.toshiba.co.uk

B. Jayakumar
Interactions, 41 Spring Street, New Providence, NJ, USA
e-mail: bjayakumar@interactions.com

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_3

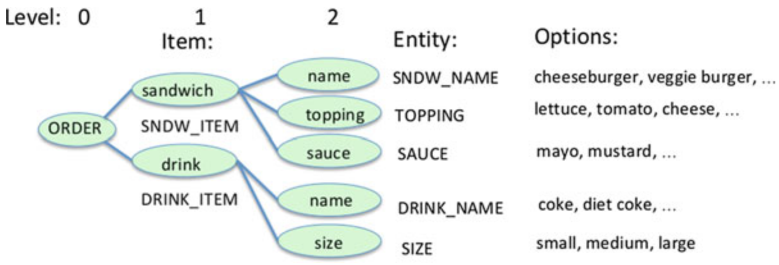


Fig. 1 Domain definition example in food ordering domain

general post-processing of the ASR/NLU method for improving interpretation in a dialog system using context, compared to methods explored in previous work [14, 17].

Figure 1 shows an abbreviated schematic KGD definition of the food ordering domain. The domain is represented with the three-level structure: task (0), subtask/form (1), and concept/entity (2), similar to [7].

The dialog manager takes as input the result of the potentially noisy natural language understanding (NLU) component. The F-measure for detecting intent, entities, and items in the food ordering dataset used in this work is 0.67/0.48 on manual/automatic transcripts [6]. As the information state update (ISU) dialog model [18], KGD maintains a symbolic information state. In the original ISU, the dialog manager performs state update based on the NLU output. The move ranking approach adds a stochastic computation to the dialog manager prior to executing the state update, aiming to overcome errors in the noisy input.

The domain representation (Fig. 1) defines the structure of the dialog information state. The dialog state maintains a list for each subtask/form (the *level 1* nodes of the domain structure) and is updated during every turn of the dialog. In the food ordering domain, the information state at each dialog turn corresponds to a partial order as items are added, removed, or modified. We introduce the notion of a *unit move*—a unit modification of a dialog system state. Unit moves include adding a new item of a particular type using one of its attributes, removing an item, or modifying an attribute value. Table 1 shows a sequence of unit moves and their resulting dialog states.

At each dialog turn, the *MoveRank* DM generates a list of valid moves for a given dialog state, such as adding a new item, removing or modifying an item currently in the dialog state, or requesting information about any of the items in the domain or the current order. Instead of directly executing the output of the NL to generate a new information state, *MoveRank* estimates the probability of each valid move based on the NLU output, and executes a set of the top-ranking moves generating a new system dialog state.

In the next sections, we describe the *MoveRank* algorithm and present the evaluation results on dialogs from the food ordering domain. We show that: (1) on the *gold NL* dataset, the stochastic component of ranking does not introduce errors;

Table 1 Information state update example with a sequence of unit moves. The initial system state is empty

Utt	Unit move	Result dialog state
can I have a cheeseburger	new_item(type=sandwich, name=cheeseburger)	sdwchs:[name=cheeseburger]
and a large drink	new_item(type=drink, size=large)	sdwchs:[name=cheeseburger]; drinks: [size=large]
add lettuce	modify(type=sandwich, topping=lettuce, index=0)	sdwchs:[name=cheeseburger, top=[lettuce]]; drinks: [size=large]
forget cheeseburger	remove_item(type=sandwich, index=0)	drinks: [size=large]

(2) on a *backed-off-gold NL* dataset with no entity and item labels, the move ranker is still correct; and (3) with the fuzzy encoding of the NL, aimed at handling spelling variations, the performance of selecting correct moves is still above 0.80.

2 Related Work

The knowledge graph driven dialog management approach is motivated by previous methods that simplify task-oriented dialog system authoring and facilitate reuse of generic components across domains [1, 4, 5, 12, 20]. The focus of the *MoveRank* approach is to overcome the problem of ASR and NLU error propagation in a knowledge driven dialog manager, and has been previously addressed in [2, 3, 8].

Unlike belief state tracking, which maintains a distribution of the dialog states [10], the proposed approach maintains a single version of the dialog state. *MoveRank* uses an unsupervised similarity measure to select *unit moves* and compute the new dialog state. Reinforcement learning is commonly used to learn the optimal system policy from data, e.g., [9, 11]. In the proposed approach, we separate the functions that update the state from those that determine the next move. The dialog manager first performs a *state update* and then determines the *system response*. The former is addressed in this paper, with a plan to apply reinforcement learning to the latter in future work.

The proposed *MoveRank* method is unsupervised. Motivated by a hybrid solution combining a Recurrent Neural Network and domain-specific knowledge [19], we envision a supervised learning approach for *MoveRank*, using a deep neural network trained to optimize a similarity measure between the move and the utterance. In our approach, however, we will be classifying moves for the state update rather than the system response. Decomposing the dialog manager into *state update* and *system response* components allows the KGD framework to support mixed-initiative systems with structured information states, such as a dictionary of lists for the shopping domain.

3 Method

A dialog system author defines the domain structure in a graph that is used to initialize a domain-independent dialog manager. Figure 1 shows a schematic domain definition in the food ordering domain. Food ordering is the task of *structured information collection*, where a domain has three levels, with task, forms, and attributes. A form node corresponds to a menu, and is static. An information state is modified during the dialog. It corresponds to a customer's order. Multiple menu items may be added to the information state whose structure is defined by the domain.

The KGD move ranking algorithm applies at each turn of the dialog. It takes as input the previous dialog state S_0, the output of the NLU component NL, and the context C. It returns a set of selected moves M_{sel} and the new dialog state $S1$:

Algorithm 1 Move Ranking Algorithm

procedure $MoveRank(S_0, NL, Context)$
  $[M_{gen}] \leftarrow MoveGen(S\_0)$ # **move generator** generates a list of symbolic moves
  $[h_m] \leftarrow EncodeMoves([M_{gen}])$ # **encoder** encodes each symbolic move into a vector
  $h_{NL} \leftarrow EncodeNL(NL)$ # **encoder** encodes symbolic NL into a vector
  $[s_m] \leftarrow MoveScorrer([h_m], h_{NL}$ # **scorer** assign a score to each move
  $[M_{sel}] \leftarrow MoveSelector([M_{gen}], s_m)$ # **selector** selects the most likely moves
  $S_1 \leftarrow MoveExecutor(S_0, [M_s el])$ # **executor** generate a new state
  return M_{sel}, S_1

The system components involved in the interpretation of a user utterance are *move generator, encoder, scorer, and selector* (Algorithm 1). The **move generator** dynamically generates a set of symbolic **user moves** $M_{gen} = \{m_1, m2, ..., m_k\}$ based on the current information state. A move m_i is a symbolic structure. It is either *state-modifying* (corresponding to a modification of a single value in an information state and referred to as a *move*) or *info-requesting* (corresponding to a request for information, such as a menu item or order status). In the food ordering domain, the domain actions include adding an item, removing an item, or changing an existing item's attribute. With an empty system state, the possible *moves* are to add any of the menu items. When a system state contains one or more items, the possible *moves* include removal or modification of any of the items in the current state. A move unambiguously describes a unit modification of a dialog state. Because M_{gen} is generated dynamically, it includes only valid moves that can be executed on the current state. A move structure includes *type* (add/remove/modify), *id* (composed of item and attribute from the domain definition), *val* (new value to set). *Modify* moves also have *mod-type* which can be set or remove (for list attribute such as topping) and *context* which describes referring attributes of the modified item. A move corresponding to adding a cheeseburger to an order (where cheeseburger is the name of a menu item) is:

type:	new_item
id (item_attr):	sandwich_name
val:	*cheeseburger*

A move corresponding to removal of lettuce from the first sandwich in the info state is:

type:	modify_attribute
id (item_attr):	sandwich_topping
index:	0
val:	*lettuce*/TOPPING
mod-type:	remove
context:	[*cheeseburger*]

Adding an item with more than one attribute, such as *cheeseburger with no lettuce*, requires both of the above moves.

We define a domain key vector $D \in \{string\}^k$ with domain functions, entity/item types, and string option values for the corresponding domain definition. The key values are derived from the domain definition and correspond to the values column in Table 2. The **encoder** uses D to convert a symbolic move and the NLU output into semantic vectors h_M and h_{NL} of length k. $h_M = \mathbf{E}_m(M, D)$, where E_m is a move encoding function applied to each symbolic move. $h_M \in \{0, 1\}^k$, is a binary vector with 1's set where D matches the move:

$$h_{M_i} = \begin{cases} 1 \iff (M.type = D_i \vee (\exists c \in M.context \wedge c = D_i) \vee M.val = D_i) \\ 0 \ otherwise \end{cases} \tag{1}$$

For example, the above move turns on the bits corresponding to '*lettuce*', 'TOPPING_*lettuce*', 'type:modify', and '*cheeseburger*', which may be used to refer to the modified item.

NLU assigns an intent tag (NL_{in}) on the whole segment, entity tags (NL_{en}) on entity strings (NL_{str}), and item grouping tags (NL_{it}), as illustrated in Table 3.

We encode the NLU output with **binary** and **fuzzy** encoding methods. The binary encoder generates $h_{NL} \in \{0, 1\}^k$, where h_{NL_i} is computed using exact match of NLU labels with the domain vector D:

Table 2 Domain function, entity/item types and option values for the food ordering domain example defined in Fig. 1

Category	Values
Domain function	add_item, remove_item, modify_attribute, request_info
Entity/item type	SNDW_NAME, TOPPING, SAUCE, DRINK_NAME, SIZE, SNDW_ITEM, DRINK_ITEM, etc.
Option value	cheeseburger, veggie burger, lettuce, tomato, cheese, mayo, mustard, coke, diet coke, small, medium, large, etc.

Table 3 User utterance annotation example

utt: *Can I have a cheeseburger without lettuce please*	
intent (NL_{in}):	add_item
entity (NL_{str}/NL_{en}):	*cheeseburger*/sandwich_name
entity (NL_{str}/NL_{en}):	*lettuce*/sandwich_topping
item: (NL_{str}/NL_{it})	*cheeseburger without lettuce*/sandwich_item

$$binary\,enc:\ h_{NL_i} = \begin{cases} 1 \iff (D_i \in NL_{en} \vee D_i \in NL_{in} \vee D_i \in NL_{it} \vee D_i \in NL_{str}) \\ 0\ otherwise \end{cases} \tag{2}$$

where NL_{en}, NL_{in}, NL_{it} are the *entity, intent,* and *item* tag names and NL_{str} are tagged *entity values*. Unlike system generated symbolic moves, the NL output may contain misspellings which will not be captured by the binary match. The fuzzy NL encoder generates a vector $h_{NL} \in \Re^k | 0 \leq_{NL_i} \leq 1$, where h_{NL_i} is computed using an exact match for intents, and entity types and edit distance for the string values.

$$fuzzy\ enc:\ h_{NL_i} = \max \begin{cases} 1 \iff (D_i \in NL_{en} \vee D_i \in NL_{in} \vee D_i \in NL_{it}) \\ \max_{S \in \{NL_{str}\}} (1 - NormEditDist(D_i, S)) \end{cases} \tag{3}$$

Move scorer computes $s_M \in \Re^k$, a vector of scores for each possible symbolic move m_i. A move score, s_{M_i}, estimates the likelihood that a move m_i was intended by the user in the utterance based on the NL output h_{NL}, and the previous system utterance context.

The **scorer** computes s_{M_i} for each move vector h_{M_i} by a dot-product with the h_{NL}:

$$s_{M_i} = (h_{M_i} \cdot h_{NL}) \cdot C_i \tag{4}$$

where C is the context vector $C \in \Re^k$:

$$C_i = \begin{cases} \gamma > 1 \iff m_i(id) = Sys\_context(id) \wedge m_i(index) = Sys\_context(index) \\ 1\ otherwise \end{cases} \tag{5}$$

$\gamma > 1$ increases the score of the moves with the matching system context, $Sys\_context = \{id, index\}$, a tuple identifying the node of the information state under discussion. For example, in the following dialog snippet,

(1) User: Can I get diet coke?
(2) System: What size?
(3) User: Large.

after (1) is processed, the dialog state contains one drink item (diet coke). The system's utterance (2) sets $Sys\_context : \{id : drink\_size, index : 0\}$, indicating that the size for the 0th drink in the dialog state is requested. When (3) is processed, the moves corresponding to setting the size of 0'th drink item in the dialog state will be scored higher than the other possible moves (such as *adding a large pizza*). We heuristically set $\gamma = 1.5$.

A probability distribution p_M over moves is computed by taking a *softmax* over the vector of all moves s_M:

$$p_M = softmax \begin{bmatrix} s_{M_1} \\ s_{M_2} \\ \dots \\ s_{M_N} \end{bmatrix} \tag{6}$$

Move selector picks a set of most likely moves M_{select} to be executed. Each input segment has a unique intent, and applies to one item.[1] Hence, for each segment we select the moves corresponding to one item. An NL segment may correspond to multiple moves, as in the "*add a cheeseburger without lettuce*" example, hence the size of the set $|M_{select}| \geq 1$.[2] We define a frame to be a grouping of moves with shared type, item, and index. For example, *frame(modify, sandwich, 0)* includes all valid moves modifying the 0'th sandwich. From the set of generated moves $m_1, m_2, ..., m_k$ and their corresponding scores $s_{m1}, s_{m2}, ..., s_{mk}$,[3] we compute a score for each frame and select the set of moves corresponding to the top ranked frame.

Move executor iteratively applies state update method to the input state S_{in} for each move in M_{sel}, to obtain the new system state.

4 Data

We manually authored a KGD domain definition for a fast-food restaurant previously used for data collection [6]. In the future, we envision automatic generation of the domain structure from a menu and a point-of-sale system.

We use 8 manually annotated dialogs with 21 user utterances to obtain the *seed gold dataset*. The utterances were split into 39 single-intent segments with 24 *add_item*, 9 *resp_yes/no*, and 6 *assert* intents. For each of the segments, we obtain a *gold* data point tuple: $< S_{in}, NL_{gold}, A_S, S_{res} >$. For the first utterance in the dialog, S_{in} is empty. NL_{gold} is the manual annotation including *intent, entity,* and *item* labels illustrated in Table 3, and described in detail in [6]. We manually annotate the agent utterances A_S with a dialog act,[4] id and/or $index$ of an item in a dialog state S_{in} referred to in the agent utterance (see Table 4). We run the MoveRank (Algorithm 1)

[1] We pre-process NL output to contain single item in each input segment based on the *item span* NL labels.

[2] For the out-of-vocabulary utterances *no match* move is selected and $|M_{select}| = 1$.

[3] We consider the moves with the scores above the empirically selected threshold T.

[4] Not currently used by the system.

Table 4 Agent annotations. DA is a dialog act, item is labeled with an id and index of the intem in info state

Example	DA	Item id	Item index
What else would you like?	General request	–	–
What size of fries would you you like?	Req info	Side_size	$< index >$
Would you like cheese on your sandwich?	Propose	Sandwich_topping	$< index >$

with the inputs S_{in}, NL_{gold}, A_S, generating the resulting state S_{res} which is used to initialize the S_{in} of the next data point. We repeat this procedure for each annotated dialog and manually confirm the correctness of each resulting state S_{res} in the *seed dataset*.

The *seed dataset* is expanded with the template expansion approach. We convert each annotated user utterance NL_{gold} into a template by replacing entity values with the entity label, and use the templates to generate all possible renderings of this template by substituting each matching option from the domain structure. For example, an utterance *"Can I get a* **diet coke** *with it"*, is converted into a template *"Can I get a DRINK_NAME with it"*. This template generates strings for each DRINK_NAME in the menu: *"Can I get a* **sprite** *with it"*, *"Can I get a* **coke** *with it"*, *"Can I get a* **iced tea** *with it"*, etc. We obtain 967 expanded inputs $< S_{in}, NL_{exp}, A_S >$, and by running the KGD on them, we generate the output state S_{res} and selected moves M_{sel}.

We verify that the resulting symbolic state S_{res} for the expanded instances of the same template are structurally equivalent: they contain the same sets of item types with varying string values. With this process, we obtain 967 data points for the *gold expanded* dataset which we use for the evaluation.

5 Evaluation

In this work we evaluate the *MoveRank* state update approach, a part of the KGD dialog management framework proposed in [15]. *MoveRank* selects *unit moves* (M_{sel}), and deterministically applies them with the state update operation to the previous dialog state, generating the new resulting state S_{res}. As the state update is deterministic, applying a correct set of moves results in a correct result state. To evaluate *MoveRank*, we compute **precision, recall,** and **F-measure** of the selected moves M_{sel}. The correct selected moves $CORR_M$ are generated by running the *MoveRank* on the *gold expanded* dataset with *binary* encoding condition. The output of each utterance in the *seed gold* dataset was manually confirmed to be correct.

The quality of move scoring affects the performance of move selection. When the moves in $CORR_M$ are ranked higher, they are more likely to be selected, leading to a higher system score. We separately evaluate the stochastic *move scorer* component using **mean reciprocal ranking** MRR_i on the sorted list of scored moves for each data point i:

$$MRR_i = \frac{\sum_{m \in CORR_M} (\frac{1}{rank(m, M_{scored})})}{|CORR_M|} \tag{7}$$

where $rank(m, M_{scored})$ is the rank of the move m in the list of *scored moves* M_{scored} sorted by their score. Moves with the equal score have the same rank. A scorer that ranks each move in $CORR_M$ as #1 would have $MRR = 1$. The final MRR score is a micro average of the MRR scores for all data points. Selecting a system response action is performed in a separate KGD component and is out of the scope of this paper.

To evaluate the *MoveRank*'s robustness to the missing information in the NLU output, we create an *expanded gold back-off* dataset by replacing with UNK (1) all entity labels, (2) all item labels, (3) entity and item labels, (4) intent labels, and (5) intent, item, and entity labels. Without the labels, the NL only identifies entity and item strings.

For an utterance *"Can I have a cheeseburger with pickle please"*, the NL output in the experimental condition (3) with no entity and item label is *add_item(type=UNK, entities: [cheeseburger/UNK, pickle/UNK)]*. The domain encoder will encode the strings corresponding to the entity values but not the missing labels.

Table 5 shows the evaluation results. *MoveRank* with the binary encoding has MRR $0.99 - 1$ and always selects correct moves on the *gold* and on all of the *gold back-off* datasets.[5] A deterministic rule-based dialog manager would perform perfectly on the *gold* dataset. However, to support the input with missing or incorrect labels would require domain-specific rules, while *MoveRank* provides this functionality generally.

The performance with the binary encoding in the conditions (1–3) without entity and item labels is perfect because the strings in this dataset correspond to the entity values or their known paraphrases, e.g. *fry* and *fries* for *french fries*.[6] The system does not rely on entity and item labels because it can match the strings with the domain entity values. Interestingly, with the binary encoding in the condition (4) and (5) with no intent labels, the *MoveRank* performance remains P=R=F=1.0. This may be due to the lack of intent variability in the small set of templates in our dataset.

To handle inevitable misspellings, we use *fuzzy* encoding (see Sect. 3). Fuzzy encoding may lead to incorrect moves receiving a higher score because of a chance match with the domain keys in the NL encoding. For example, *'iced coffee'* would have a partial match with *'hot coffee'* which may lead to invalid ranking and move selection. With the fuzzy encoding, the MRR on the *gold* NL annotations drops to

[5] We experimentally picked a threshold for move selection with binary encoding $T = 0.5$.

[6] We add the paraphrases encountered in the data to the domain.

Table 5 Evaluation of move scoring (MRR) and move selection (P,R,F) for the unsupervised *MoveRank*

NL annotation	Encoding	MRR	P	R	F
Gold	Binary	0.99	1.0	1.0	1.0
(1) No entity	Binary	1.0	1.0	1.0	1.0
(2) No item	Binary	0.99	1.0	1.0	1.0
(3) No entity/item	Binary	1.0	1.0	1.0	1.0
(4) No intent	Binary	0.99	1.0	1.0	1.0
(5) No entity/item/intent	Binary	1.0	1.0	1.0	1.0
Gold	Fuzzy	0.78	0.83	0.89	0.86
(1) No entity	Fuzzy	0.78	0.84	0.80	0.82
(2) No item	Fuzzy	0.78	0.83	0.89	0.86
(3) No entity/item	Fuzzy	0.78	0.84	0.80	0.82
(4) No intent	Fuzzy	0.70	0.83	0.80	0.81
(5) No entity/item/intent	Fuzzy	0.73	0.83	0.68	0.75

0.78. Note that the absence of entity and item labels (conditions 1–3) does not affect the MRR score. Removing intent labels reduces the MRR to 0.70. The F-measure for the move selection with fuzzy encoding is 0.86 on the *gold* dataset and on the dataset with no item labels (3). Removing enity labels reduces F-measure to 0.82 in the conditions (1) and (3). And with no labels (5), the performance drops to $F = 0.75$.

6 Discussion

In this work we presented *MoveRank*, a novel approach to an information state update dialog manager. *MoveRank* applies in the framework of a knowledge graph driven dialog manager (KGD), extending the idea of system generation from a form [16]. With the KGD framework and using mostly a generic code base, a fully functional dialog manager is created declaratively.

MoveRank is a novel approach for computing the contextual resolution and disambiguation of a user's intent. It combines symbolic and continuous representations: the domain and the state are represented symbolically, while the moves are scored using a continuous vector representation. One of the drawbacks of a rule-based system is the ASR and NLU error propagation to the dialog manager. *MoveRank* provides an elegant solution for the cascading ASR/NLU errors. Fuzzy encoding allows efficient combination of multiple N-best hypotheses and uses partial string matches. The moves are scored in the presence of ASR errors, misspelled words and NLU errors, and correct moves may be selected.

We described a baseline unsupervised approach, where move scores are estimated by semantic closeness between the move and NL, and computed using a dot-product.

In future work, we will experiment with statistical models by training the scoring component on the data. A statistical *MoveRank* will combine complementary strengths of knowledge-driven and statistical approaches. Furthermore, since user utterances were not used for the scoring in the baseline approach, a future approach could use contextual embedding like ELMo [13] for the utterances, which has shown significant improvement in many downstream applications.

References

1. Allen J, Byron D, Dzikovska M, Ferguson G, Galescu L, Stent A (2000) An architecture for a generic dialogue shell. Nat Lang Eng 6(3–4):213–228
2. Bohus D (2007) Error awareness and recovery in conversational spoken language interfaces. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA
3. Bohus D, Langner B, Raux A, Black A, Eskenazi M, Rudnicky A (2006) Online supervised learning of non-understanding recovery policies. In: Proceedings of IEEE Workshop on Spoken Language Technology, SLT
4. Bohus D, Rudnicky A (2003) Ravenclaw: dialog management using hierarchical task decomposition and an expectation agenda. In: Proceedings of Eurospeech
5. Bos J, Klein E, Lemon O, Oka T (2003) DIPPER: description and formalisation of an information-state update dialogue system architecture. In: proceedings of the 4th annual meeting of the special interest group on discourse and dialogue, SIGDIAL, pp 115–124
6. Chen J, Prasad R, Stoyanchev S, Selfridge E, Bangalore S, Johnston M (2018) Corpus and annotation towards nlu for customer ordering dialogs. In: Proceedings of IEEE Workshop on Spoken Language Technology, SLT. Athens, Greece
7. Chotimongkol A, Rudnicky A (2008) Acquiring domain-specific dialog information from task-oriented human-human interaction through an unsupervised learning. In: Proceedings of the 2008 conference on empirical methods in natural language processing, pp 955–964. Association for Computational Linguistics, Honolulu, Hawaii
8. Bohus D, Rudnicky A (2005) Sorry, i didn’t catch that!—an investigation of non-understanding errors and recovery strategies. In: Proceedings of the 6th annual meeting of the special interest group on discourse and dialogue, SIGDIAL
9. Henderson J, Lemon O, Georgila K (2006) Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. Comput Linguist 34:487–511
10. Henderson M, Thomson B, Williams JD (2014) The second dialog state tracking challenge. In: Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue, SIGDIAL, pp 263–272
11. Lemon O, Georgila K, Henderson J (2006) Evaluating effectiveness and portability of reinforcement learned dialogue strategies with real users: the Talk TownInfo evaluation. In: Proceedings of IEEE workshop on spoken language technology, SLT
12. Lison P, Kennington C (2016) OpenDial: a toolkit for developing spoken dialogue systems with probabilistic rules. In: Proceedings of the 54th annual meeting of the association for computational linguistics (Demonstrations), pp 67–72
13. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Proceedings of NAACL
14. Stoyanchev S, Johnston M (2015) Localized error detection for targeted clarification in a virtual assistant. In: Proceedings of IEEE international conference on acoustics, speech and signal processing, ICASSP, pp 5241–5245
15. Stoyanchev S, Johnston M (2018) Knowledge-graph driven information state approach to dialog. In: Proceedings of AAAI workshops on reasoning and learning for human-machine dialogues

16. Stoyanchev S, Lison P, Bangalore S (2016) Rapid prototyping of form-driven dialogue systems using an open-source framework. In: Proceedings of the 17th annual meeting of the special interest group on discourse and dialogue, SIGDIAL
17. Stoyanchev S, Salletmayr P, Yang J, Hirschberg J (2012) Localized detection of speech recognition errors. In: Proceedings of IEEE workshop on spoken language technology, SLT, pp 25–30
18. Traum D, Larsson S (2003) The information state approach to dialogue management. In: Current and new directions in discourse and dialogue, pp 325–353. Springer, Dordrecht, Netherlands
19. Williams JD, Asadi K, Zweig G (2017) Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In: Proceedings of the 55th annual meeting of the association for computational linguistics, ACL
20. Xu W, Rudnicky AI (2000) Task-based dialog management using an agenda. In: Proceedings of the 2000 ANLP/NAACL workshop on conversational systems—volume 3, ANLP/NAACL-ConvSyst '00, pp 42–47

Predicting Laughter Relevance Spaces in Dialogue

Vladislav Maraev, Christine Howes, and Jean-Philippe Bernardy

Abstract In this paper we address the task of predicting spaces in interaction where laughter can occur. We introduce the new task of predicting actual laughs in dialogue and address it with various deep learning models, namely recurrent neural network (RNN), convolution neural network (CNN) and combinations of these. We also attempt to evaluate human performance for this task via an Amazon Mechanical Turk (AMT) experiment. The main finding of the present work is that deep learning models outperform untrained humans in this task.

1 Introduction

Non-verbal vocalisations, such as laughter, are ubiquitous in our everyday interactions. In the Switchboard Dialogue Act corpus [7], which we use in the current study, non-verbal dialogue acts (that are explicitly marked as non-verbal) constitute 1.7% of all dialogue acts and laughter tokens make up 0.5% of all the tokens that occur in the corpus. In order to make state-of-the-art dialogue systems more natural and cooperative, it is vital to enable them to understand non-verbal vocalisations and react to them appropriately. With regards to laughter, the most important issues are in understanding the coordination of laughter with speech, social and pragmatic functions of laughter, and the reasons for laughter.

The social function of laughter is well documented e.g., [10]: laughter is associated with senses of closeness and affiliation, establishing social bonding and smoothing

V. Maraev (✉) · C. Howes · J.-P. Bernardy
Department of Philosophy, Linguistics and Theory of Science, Centre for Linguistic Theory and Studies in Probability (CLASP), University of Gothenburg, Gothenburg, Sweden
e-mail: vladislav.maraev@gu.se

C. Howes
e-mail: christine.howes@gu.se

J.-P. Bernardy
e-mail: jean-philippe.bernardy@gu.se

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_4

away discomfort. Laughter can also have a pragmatic function, such as indicating a mismatch between what was said and what was meant, for example by indicating that a speaker was ‘just kidding’ (for further details on classification of laughter see the work of [9]).

Although laughter is closely associated with humour, and humorous and joyful remarks can be thought as a prerequisite for laughter, this is not necessarily the case: laughter can display surprise, nervousness, embarrassment, disagreement etc. [13]. This suggests that laughter is not exclusively associated with positive emotions (happiness, joy, pleasure and more)—other emotional dimensions and their (perhaps contradictory) combinations should also be considered. Nevertheless, positive emotional states are an intuitive notion of where laughter occurs.

In the current study we focus on the issues of laughter relevance and predictability. We introduce the term *laughter relevance spaces* analogously with backchannel relevance spaces [4] and transition relevance places [14]. We define a *laughter relevance space* as a position within the interaction where an interlocutor can appropriately produce a laughter (either during their own or someone else’s speech). Following the approach of [4] to backchannels, we distinguish *actual laughs* and *potential laughs*. By definition, the number of potential spaces for laughter is larger than number of actually produced laughter spaces.

In this work we are guided by the following research questions: (i) can laughs be predicted from the textual data either by humans or by deep learning systems, and (ii) to what extent can these predictions be compared. In an attempt to address these questions we present:

- The task of predicting laughter from dialogue transcriptions.
- Human annotations of potential laughs from dialogue transcriptions.
- Automatic methods for predicting actual laughs with deep learning models.

In the rest of the paper we present details of the dataset and the task (Sect. 2). We then describe the Amazon Mechanical Turk (AMT) experiment and its evaluation (Sect. 3). We present our sentiment analysis baseline in Sect. 4. In Sect. 5 we present our deep learning models and summarise the results. We conclude with some pointers for future work (Sect. 7).

2 Data

The Switchboard Dialogue Act Corpus [7] consists of 1155 dyadic telephone conversations (221,616 utterances) between participants who were unfamiliar to each other. For the purposes of our study we make use of the disfluency annotations [11] in the corpus.

For our experiments we split utterances into tokens using the Python library SWDA[1] and combine consecutive laughs within a turn into a single laughter token.

[1] https://github.com/cgpotts/swda.

Fig. 1 Example of dialogue split into two samples. The leading number shows to which of the training samples each utterance will be related to based on 3-turn span

```
1 sp_A {F Oh, } I know. /
1 sp_A It's really amazing. /
1 sp_B Yeah. /
2 sp_A It's, {F uh, } <LAUGHTER> -/
2 sp_B Beautiful, beautiful machine. /
2 sp_A Absolutely, /
```

Table 1 Predicted laughs depending on a turn span and threshold. Number of laughs to predict vary due to different splits of the data

Span	Threshold	Laughs to predict	Precision	Recall	F_1
3	0.50	1128	0.733	0.010	0.007
5	0.50	1116	0.786	0.010	0.005
10	0.50	1127	0.630	0.015	0.018
10	0.45	1127	0.407	0.020	0.132
10	0.40	1127	0.400	0.039	0.036
10	0.35	1127	0.255	0.060	0.049

The laughter tokens are then removed from the text and replaced by laughter annotations. That is, the data is a sequence of tuples (t_i, l_i) such that:

- $t_i \in \mathbb{N}$ is the ith speech (typically a representing a word) or turn-taking token (For either A or B).
- $l_i \in \{0, 1\}$ is a laughter marker, which indicates whether laughter follows immediately after the token t_i.

The goal of the current study is to determine whether l_i can be predicted, that is, does laughter occur after a given sequence of tokens $(t_0..t_i)$.

Exploratory Task
The obvious way to tackle the goal is to predict the probability of laughter for each token. To do so we split the corpus on turn boundaries, with no overlap (Fig. 1) and train an RNN model (see Sect. 5.1.1) on 80% of the corpus (total number of samples range from 17k examples for 10-turn split to 73k for 3-turn split). In Table 1 we report the results depending on a turn span and threshold for converting predicted probability of laughter into a binary value. We observed that adding more context leads to better predictions even if it leads to decreasing the size of training data.[2]. Yet, even in the case of a 10-turn span, the recall was only 1.5%. A direct attempt to increase the recall by increasing the threshold to report a laughter lowered the precision to unacceptable levels.

Balanced Task
The above experiment indicates to us that this task is difficult to tackle using deep learning models. We attribute this difficulty to the corpus being unbalanced towards

[2]In all our experiments we keep 80%/10%/10% training/validation/test split.

negative predictions, due to the sparsity of laughs. Indeed, the proportion of actual laughter tokens is around 0.5% in the whole corpus. Additionally, it is also a hard and unrealistic task for humans because annotating every token is tedious.

We therefore, instead, fix the point of focus to given positions, and attempt to predict the incidence of laughter those given points. We select these points so that the frequency of laughter is equal to the frequency of non-laughter at this points. To do so, we run a sliding window through all the examples. The size of the sliding window is fixed to a given number of tokens (not turns), in our case, 50 or 100 tokens. All the laughs (except the final one for the sequence) are represented as a special token and the final laughter is removed and represented as a positive label. The resulting training set (80% of all data) contained around 17k samples and remaining 20% were left out for validation and testing. This amended task is the focus of the rest of the paper.

3 Amazon Mechanical Turk

In order to understand how well humans perform at this task, we conducted an experiment with naive annotators located in the US. They were given a task description with the following salient points:

1. An invitation to complete the task with a notice that native level of English is required.
2. A sound sample of a dialogue containing laughs (in order to help coders understand that laughter can occur in non-humorous conditions).
3. Three excerpts from test set with removed non-verbal signals, disfluency and discourse annotations. Each of the excerpts has to be annotated regarding the potential to elicit laughter as: (a) very unlikely, (b) not very likely, (c) quite likely, and (d) very likely.

The subset of 399 excerpts was annotated by at least two annotators per sample. We computed Cohen's κ chance-adjusted inter-annotator agreement both for four-class predictions and for predictions converted into binary: judgements "quite likely" or "very likely" are counted as positive and "very unlikely" or "not very likely"—as negative. The resulting κ was very low (below chance level: $\kappa = -0.125$ for four-class predictions and $\kappa = -0.071$ for binary predictions), which indicates either that quality of AMT annotations are very low or that human judgements about laughter are very subjective.

Subjects also showed a disposition towards laughter: 66% of excerpts were annotated as "quite likely" or "very likely", and only 2% were annotated as "very unlikely" or "not very likely" by both annotators. After comparison with the distribution of actual laughs in the corpus we observe that AMT respondents are not very good in predicting whether there was actually a laughter at the end of the sequence, but they might instead be predicting *potential* laughter, which is suggested by the predominance of such predictions. This means that participants are judging whether a

Table 2 Human annotations as compared with the test set. Scores are computed based on the valence. For all the cases examples labelled as “quite likely” or “very likely” valence is positive, and the rest—as negative

Selection principle	Accuracy	Precision	Recall	F_1
Average of 4-class annotations	0.51	0.50	0.92	0.65
Average of binary annotations	0.51	0.49	0.67	0.57
Agreed judgements w.r.t. the valence (in 271 cases out of 499)	0.51	0.49	0.98	0.66

laugh could appropriately be produced at that point in the dialogue, not whether the dialogue participants themselves actually did produce one. We conjecture that this result extends to the general population, if asked to make laughter judgements in the same conditions (i.e. when little effort is spent for each judgement). The expert prediction of laughter, that is by subjects trained in the relevant psycholinguistic aspects, is beyond the scope of the present paper. In Table 2 we show accuracy and F_1 score for human predictions of actual laughs.

4 Off-the-Shelf Sentiment Analyser

Even though laughter can be associated with a variety of sentiments, it is often naively associated with positive sentiment. Therefore, as a baseline, we employed the VADER sentiment analyser [3] to check whether its prediction of positive sentiment correlates with laughter. VADER is designed to classify sentiment along the positive/negative scale and mainly used for sentiment classification in social media which is not specifically designed for the dialogue task but arguably should perform relatively well on “noisy” texts such as those found in the Switchboard corpus [5]. VADER is built in the Python NLTK library [1].

The sentiment analyser showed a predominance towards positive sentiment (and hence laughter) but the accuracy was only slightly above the majority vote baseline (51.1%).

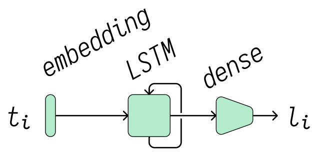


Fig. 2 Architecture of the RNN model ("rolled" view). For the main task only the final prediction (l_n) is considered

5 Deep Learning

5.1 Models

We present several deep learning models to tackle our task, either recurrent neural networks (RNN), convolutional neural networks (CNN) or combinations of these.

These models are implemented using our own high-level deep-learning library, which uses TensorFlow as a backend.[3,4]

5.1.1 RNN Model

Our RNN-based model architecture is shown in Fig. 2 and consists of three layers:

1. **An Embedding layer** which is characterised by the size of token embeddings (d).
2. **An LSTM recurrent layer** characterised by state size n. Each LSTM cell additionally includes dropout (on its inputs, outputs and hidden state inputs) of a probability ϵ.
3. **A Dense layer** which predicts laughter relevance for each token. We have exactly two classes: relevant (1) or irrelevant (0). For the main task we only output the final prediction of the dense layer.

5.1.2 CNN Model

The convolution neural network model includes the following parts:

1. **An Embedding layer** which is characterised by size of token embeddings (d).
2. **A first 1-D Convolution layer** characterised by filter size h_1 and number of filters k_1. The layer is followed by a rectified linear unit (ReLU).
3. **A first max-pooling layer** with a stride $s = 2$.
4. **A second 1-D Convolution layer** characterised by filter size h_2 and number of filters k_2. The layer is followed by ReLU.

[3]TypedFlow: https://github.com/GU-CLASP/TypedFlow.

[4]Models and data are available at: https://github.com/GU-CLASP/laughter-spaces.

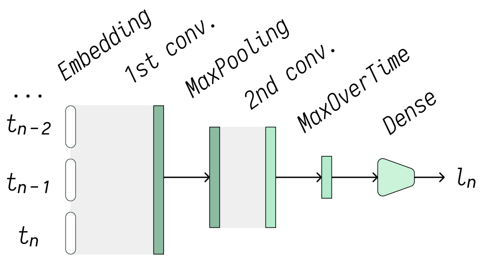


Fig. 3 Architecture of the CNN model

5. **A max-over-time pooling layer** which computes element-wise maximum along all the features of the second convolution layer.
6. **A Dense layer** that predicts laughter relevance for the sequence (Fig. 3).

5.1.3 Combinations of the Models

In order to estimate whether RNN and CNN models pick up on either the same or different features, we also tested two combinations of the above models:

1. **A Fusion model** (Fig. 4) where outputs of an RNN and a CNN model (both without a dense layer) are concatenated, and a dense layer operates on this concatenation.
2. **A Hybrid model** (Fig. 5) similar to the fusion model, but when token embeddings are shared between RNN and CNN.

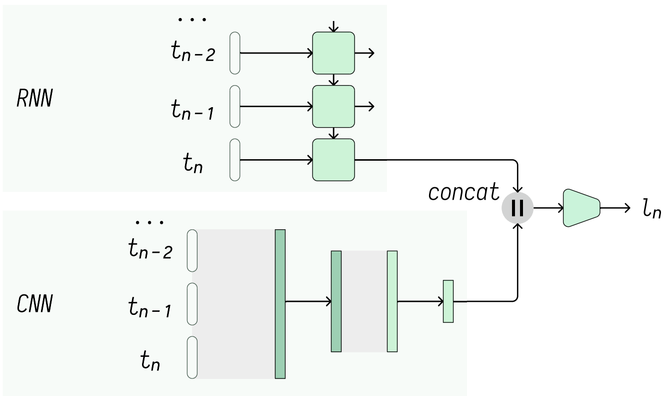


Fig. 4 Architecture of the fusion model. Outputs of the RNN's last cell and CNN's max-over-time pooling layers are concatenated and then dense layer is applied

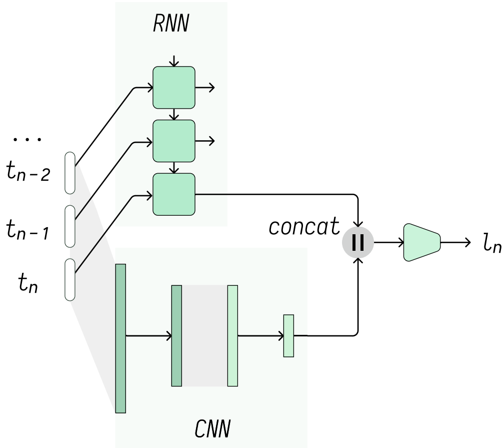


Fig. 5 Architecture of the hybrid model. Token embeddings are shared between RNN and CNN

5.2 *Results*

We present results for the different models in Table 3. Given the results of the AMT experiment, we posited that the task of predicting actual laughters in dialogue is hard for untrained humans to perform. We also saw that the task is difficult to tackle by simple sentiment analysis. Thus we expect the task to be difficult for deep learn-

Table 3 Summary of the prediction results

Model	Val. acc.	Test. acc.	Test. precision	Test. recall	Test. F_1
AMT	–	0.510	0.500	0.920	0.650
VADER	–	0.518	0.511	0.749	0.607
RNN (span = 50)[a]	0.762	0.743	0.732	0.763	0.747
RNN (span = 100)[a]	0.756	0.770	0.761	0.777	0.769
CNN (span = 50)[b]	0.789	0.765	0.761	0.771	0.766
CNN (span = 100)[b]	0.783	0.787	0.777	0.794	0.785
Fusion (span = 50)[c]	0.794	0.766	0.760	0.778	0.768
Hybrid (span = 50)[d]	0.793	0.776	0.775	0.774	0.774

Hyperparameters:
[a] $d = 50;\ n = 40;\ \epsilon = 0.1$
[b] $d = 100;\ k_{1,2} = 40;\ h_{1,2} = 7$
[c] see LSTM and CNN
[d] see LSTM and CNN, shared embedding layer: $d = 100$

ing models as well. However, the deep learning models perform considerably better than our baselines, especially in terms of accuracy. Additionally, the CNN model consistently outperforms the RNN model. Further, combining RNN with CNN provides no significant benefit. This suggests that the RNN model does not detect more laughter-inducing patterns than the CNN model.

6 Error Analysis

After analysing the results, we noted that there were a large number of examples where laughter occurs at a turn boundary. In this case the last token of the sample is a turn change (TC) token (`sp_A` or `sp_B`). A concern was that this would significantly affect the results. In order to measure this effect, we removed these results from the test set and observed the accuracy and F-measure shown in Table 4. We observe a drop of F-score (around 6 percentage points) but accuracy is almost unchanged. This indicates that system relies on turn change (possibly combined with other features—and consequently captured by neural networks) as an important predictor for laughter, for both basic models of the system. Examples where laughter is predicted to occur immediately after a turn change are shown in (1) and (2).

(1) A A: let me ask you this.

A: How, how old are you?
B: I'm, uh, thirty-three.
A: Thirty-three?
B: Thirty-two,
B: excuse me.
A: Okay.
B: ((correct prediction: LAUGHTER))

Table 4 Performance of the models before and after removing the examples where turn change token is the last token. As a result, the dataset is 22% smaller and it is missing 36% of positive examples. All deep learning models use the dataset with the span of 50 tokens

Model	Accuracy	Precision	Recall	F_1
AMT	0.500	TBD	TBD	0.660
VADER	0.518	0.511	0.749	0.607
RNN	0.743	0.732	0.762	0.747
RNN (last TC removed)	0.738	0.673	0.705	0.689
CNN	0.765	0.761	0.771	0.766
CNN (last TC removed)	0.761	0.715	0.694	0.705

(2) A B: when I was a freshman in college

A: Uh-huh.
B: uh, my degree was in computer, uh, technology originally
B: and it seemed like it would,
B: ((wrong prediction: LAUGHTER))

In conversational analysis studies many laughs are considered to form adjacency pairs with prior laughs [6], and preceding laughter by another speaker seem to be a relevant feature for our models (e.g., (3)). However, in excepts where there are a lot of laughs the system sometimes gets it wrong (e.g., (4)).

(3) A: I'm not really sure what the ((LAUGHTER))

B: Yeah,
B: really,
B: it's one of those things that you read once,
B: and then, if you're not worried about it, you just forget about it ((LAUGHTER))
A: ((correct prediction: LAUGHTER))

(4) A: (...) don't get a hot tub and

B: ((LAUGHTER)) Yes.
A: shave my legs, I'm going to die ((LAUGHTER))
A: And I had ((LAUGHTER))
B: Yes
B: I understand that ((LAUGHTER))
A: I got enough of it right ((wrong prediction: LAUGHTER))

7 Conclusions and Future Work

The main conclusion of our experiments is that for the given task deep learning approaches perform significantly better than untrained humans.

We are optimistic that the introduced task and the approaches that we have developed are a big step towards inferring appropriate spaces for laughter from textual data. Together with approaches based on audio components e.g. [2] this should enable future dialogue systems to understand when is it appropriate to laugh. Nevertheless, we are aware of the fact that this requires understanding laughter on a deeper level, including its various semantic roles and pragmatic functions (see [8] for a discussion about integrating laughter in spoken dialogue systems).

We are planning to extend our Amazon Mechanical Turk experiments by introducing more annotators in order to get more consistent results. We are going to introduce probabilistic annotations in our future crowdsourced experiments following e.g. [12].

Regarding the task itself, we are planning to address it in a more "dialogical" way. We will consider the data not as one input to a neural network which contains speaker tokens but as two possibly overlapping streams. This will introduce the notion of coordination between speakers into the prediction model. Two streams can be also extended by additional information provided in separate inputs, such as information about disfluencies, discourse markers, fundamental frequency and other acoustic features. We are planning to see what features will make a more robust contribution to the task of predicting relevant laughs.

References

1. Bird S, Loper E (2004) NLTK: the natural language toolkit. In: Proceedings of the ACL 2004 on interactive poster and demonstration sessions. Association for Computational Linguistics, p 31
2. El Haddad K, Çakmak H, Dupont S, Dutoit T (2017) Amused speech components analysis and classification: towards an amusement arousal level assessment system. Comput Electr Eng 62:588–600
3. Gilbert CH, Hutto E (2014) Vader: a parsimonious rule-based model for sentiment analysis of social media text. In: 8th international conference on weblogs and social media (ICWSM-14)
4. Heldner M, Hjalmarsson A, Edlund J (2013) Backchannel relevance spaces. In: Nordic Prosody XI, Tartu, Estonia, 15–17 Aug 2012. Peter Lang Publishing Group, pp 137–146
5. Howes C, Purver M, McCabe R (2014) Linguistic indicators of severity and progress in online text-based therapy for depression. In: Proceedings of the workshop on computational linguistics and clinical psychology: from linguistic signal to clinical reality, pp 7–16
6. Jefferson G, Sacks H, Schegloff EA (1987) Notes on laughter in the pursuit of intimacy
7. Jurafsky D, Shriberg E, Biasca D (1997) Switchboard dialog act corpus. Technical report, International Computer Science Institute, Berkeley, CA
8. Maraev V, Mazzocconi C, Howes C, Ginzburg J (2018) Integrating laughter into spoken dialogue systems: preliminary analysis and suggested programme. In: Proceedings of the FAIM/ISCA workshop on artificial intelligence for multimodal human robot interaction, pp 9–14
9. Mazzocconi C, Maraev V, Howes C, Ginzburg J (2018) Analysis of laughables: a preliminary perception study. In: Proceedings of the workshop on dialogue and perception, pp 43–47
10. Mehu M (2011) Smiling and laughter in naturally occurring dyadic interactions: relationship to conversation, body contacts, and displacement activities. Hum Ethol Bull 26(1):10–28
11. Meteer MW, Taylor AA, MacIntyre R, Iyer R (1995) Dysfluency annotation stylebook for the switchboard corpus. University of Pennsylvania, Philadelphia
12. Passonneau RJ, Carpenter B (2014) The benefits of a model of annotation. Trans Assoc Comput Linguist 2:311–326
13. Poyatos F (1993) Paralanguage: a linguistic and interdisciplinary approach to interactive speech and sounds, vol 92. John Benjamins Publishing, Amsterdam
14. Sacks H, Schegloff EA, Jefferson G (1978) A simplest systematics for the organization of turn taking for conversation. Studies in the organization of conversational interaction. Elsevier, Amsterdam, pp 7–55

Transfer Learning for Unseen Slots in End-to-End Dialogue State Tracking

Kenji Iwata, Takami Yoshida, Hiroshi Fujimura, and Masami Akamine

Abstract This paper proposes a transfer learning algorithm for end-to-end dialogue state tracking (DST) to handle new slots with a small set of training data, which has not yet been discussed in the literature on conventional approaches. The goal of transfer learning is to improve DST performance for new slots by leveraging slot-independent parameters extracted from DST models for existing slots. An end-to-end DST model is composed of a spoken language understanding module and an update module. We assume that parameters of the update module can be slot-independent. To make the parameters slot-independent, a DST model for each existing slot is trained by sharing the parameters of the update module across all existing slots. The slot-independent parameters are transferred to a DST model for the new slot. Experimental results show that the proposed algorithm achieves 82.5% accuracy on the DSTC2 dataset, outperforming a baseline algorithm by 1.8% when applied to a small set of training data. We also show its potential robustness for the network architecture of update modules.

1 Introduction

Dialogue state tracking (DST) is one of the key components of spoken dialogue systems. A DST model estimates user intent as a dialogue state from the course of a conversation. The dialogue state is constrained by a domain *ontology*, which

K. Iwata (✉) · T. Yoshida · H. Fujimura · M. Akamine
Media AI Laboratory, Corporate Research & Development Center, Toshiba Corporation, Kawasaki-shi, Japan
e-mail: kenji4.iwata@toshiba.co.jp

T. Yoshida
e-mail: takami.yoshida@toshiba.co.jp

H. Fujimura
e-mail: hiroshi4.fujimura@toshiba.co.jp

M. Akamine
e-mail: masa.akamine@toshiba.co.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_5

describes a collection of slots (e.g, food) and their slot values (e.g, Italian, Chinese, etc. for the food slot). The system determines a subsequent action from the dialogue state.

The DST model relies on a spoken language understanding (SLU) module, which estimates turn-level user goals. A large number of DST models treat the SLU module as a separate problem [1–5]. These models mainly focus on an update module that merges user goals up to the current turn and updates the dialogue state. However, these models need much annotated training data for both DST and SLU. Thus, end-to-end methods that jointly optimize the SLU module and the update module using only annotated data for DST by deep learning have been appearing recently [6–14].

A major drawback of end-to-end DST models is that they cannot handle slots and values not included in the ontology and training data. Handling these unseen slots and values is important for real-world dialogue systems, where slots and values are often changed. For this reason, DST models require a large set of additional training data. Some DST models [6–9] treat unseen slots and values by using *delexicalisation*, which replaces words corresponding to slot names and values with general features. However, this replacement requires a list of synonyms for those names and values, and this list must be crafted by system designers with rich linguistic knowledge. Therefore, it is unrealistic to deploy such an approach for large real-world dialogue systems.

Other DST models that treat unseen slot values without using human resources have been proposed. One approach uses features of a user utterance and a system response as well as a slot name and a value as inputs of DST, and estimates whether the specified slot value is included in a dialogue state [11, 12, 14]. Another approach extracts words corresponding to slot values in a user utterance by referring to contexts derived by an attention mechanism [10, 12]. However, these models do not treat unseen slots.

This paper proposes a transfer learning algorithm for end-to-end DST to treat unseen slots with a small set of training data. In this transfer learning, we aim to improve DST performance for the new slot by leveraging slot-independent parameters extracted from the training data of existing slots. Exploiting all DST parameters trained from existing slots is probably insufficient, because there are some slot-dependent parameters in the DST model. Most important in this approach is which parameters are slot-independent.

In this paper, we assume that parameters of the update module can be slot-independent, because the update module was realized using slot-independent rules in previous literature [1]. The slot-independent parameters of the update module are obtained by training a DST model for each existing slot, and sharing the parameters of the update module across all existing slots. We transfer the trained slot-independent parameters to a DST model for the new slot, then optimize the model using training data for the new slot. We evaluated the proposed transfer learning using a fully data-driven end-to-end DST model based on [10].

The remainder of this paper is organized as follows: Sect. 2 describes related work on transfer learning. Sect. 3 presents our proposed transfer learning algorithm. Sect. 4 shows the DST network architecture used for the proposed transfer learning. Sect. 5 presents experimental results and discussion, and Sect. 6 concludes the paper.

2 Related Work

Transfer learning [15] generates a model for a target task by leveraging domain-independent knowledge learned from a source dataset. Transfer learning has been widely used in many natural language processing tasks [16], including machine translation [17], questioning and answering [18], and named entity recognition [19]. We exploit transfer learning for DST, and the goal is to improve DST performance of new slots by using slot-independent parameters learned from existing slots.

Some transfer learning approaches for SLU and DST have been proposed. Jeong and Lee [20] proposed a multi-domain SLU model that is adaptable to new domains by partitioning parameters for the SLU model into domain-dependent and domain-independent factors. Kim et al. [21] derived a representation for each slot label by canonical correlation analysis, and used label embedding for mapping label types across different domains. This mitigated the problem that the label set was variant in each domain and enabled use of a number of transfer learning techniques. These approaches are used for new domains and new slots. However, they only focus on SLU. Our algorithm can be used for new slots by exploiting the characteristics of end-to-end DST.

Mrkšić et al. [8] proposed a multi-domain DST that can be extended to achieve transfer learning between domains. First, a multi-domain DST model was trained by tying the DST parameters of all slots and using all training data. Then slot-specialized models were trained by replicating the multi-domain DST parameters for each slot and using only the slot-specific training data. This approach has the potential to enable the application of transfer learning for new slots. However, all training data should be delexicalised for training the multi-domain DST, which requires human resources to replace words corresponding to slot names and values with general features. Our algorithm is fully data-driven and does not require human input.

Zhong et al. [13] proposed a global-locally self-attentive dialogue state tracker. This DST model has slot-independent global parameters and slot-dependent local parameters in an SLU module to improve the performance of rare dialogue states. This approach can be extended to transfer learning to handle unseen values. However, it is difficult to treat new slots, because not all features are equally relevant for different slots when encoding a user utterance. Our algorithm uses slot-independent update module parameters, which allows the DST model to handle new slots.

As a similar transfer learning approach, Rastogi et al. [5] proposed scalable multi-domain DST, which comprised a multi-domain SLU module [22] and a slot-independent update module. This method shares parameters of the update module across all existing slots, and transfers the parameters to new slots. One difference between their approach and ours is whether the SLU module and the update module are trained individually or jointly. This paper is first trial of applying the transfer learning approach to end-to-end DST.

3 Proposed Transfer Learning

The goal of transfer learning in this paper was to improve DST performance for new slots with a small set of training data by leveraging slot-independent parameters extracted from a large set of training data for existing slots. The key point is which parameters are trained as slot-independent.

An end-to-end DST model is composed of an SLU module and an update module. The SLU module extracts slot values from a user utterance by referring to features of the values themselves or the contexts of those values. Slot value features obviously depend on each slot, and, for all intents and purposes, so do the context features (e.g, it is common for a user to say "I want to eat *Italian* [food slot]," but rare for a user to say "I want to eat *west* [area slot]"). Therefore, it is impossible to treat SLU parameters as slot-independent.

The update module combines the SLU output and the dialogue state of a previous turn, then outputs a dialogue state for the current turn. As a typical example of this operation, if the SLU module extracts a slot value, the update model employs this value. Another example is that if the previous state has a slot value and the SLU module does not extract any slot values, the update model retains the slot value of the previous state. There are no slot-dependent features to operate these updates. Therefore, we treat parameters of the update module as slot-independent, then apply transfer learning.

Figure 1 shows the procedure of the proposed transfer learning. First, we use the training data of existing slots $D = \{D_1, D_2, \ldots, D_C\}$, where C is the number of existing slots, and update model parameters for each SLU module $\theta^{\mathrm{s}} = \{\theta^{\mathrm{s}}_1, \theta^{\mathrm{s}}_2, \ldots, \theta^{\mathrm{s}}_C\}$ and model parameters for an update module θ^{u} by minimizing an objective function as follows:

$$L(D; \theta^{\mathrm{s}}, \theta^{\mathrm{u}}) = \sum_{c=1}^{C} L(D_c; \theta^{\mathrm{s}}_c, \theta^{\mathrm{u}}), \tag{1}$$

where $L(D_c; \theta^{\mathrm{s}}_c, \theta^{\mathrm{u}})$ denotes an objective function calculated by using training data D_c and model parameters $\theta^{\mathrm{s}}_c, \theta^{\mathrm{u}}$. Note that model parameters for the update module θ^{u} are shared across all existing slots.

After training, we obtain model parameters for the update module $\theta^{\mathrm{u}*}$ that minimize objective function $L(D; \theta^{\mathrm{s}}, \theta^{\mathrm{u}})$. Next, model parameters for a new slot are trained using training data for the new slot D_{new}. $\theta^{\mathrm{u}*}$ is used for the initial parameters of the update module, then an objective function is defined as $L_{\mathrm{new}}(D_{\mathrm{new}}; \theta^{\mathrm{s}}_{\mathrm{new}}, \theta^{\mathrm{u}*})$, where $\theta^{\mathrm{s}}_{\mathrm{new}}$ is model parameters for the SLU module.

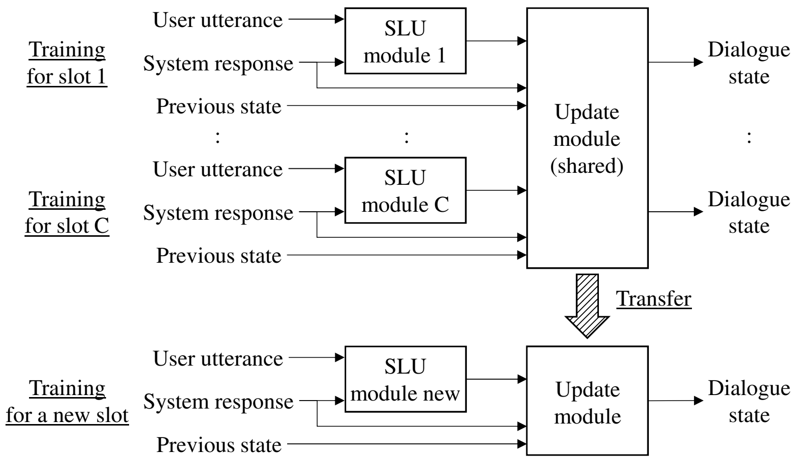


Fig. 1 Proposed transfer learning procedure. A system response is used as an input for the SLU and/or the update module depending on the network architecture

4 DST Architecture

As the end-to-end DST model for the proposed transfer learning, we prepared an attention-based SLU and update modules inspired by the extended attention-based DST model [10], which is a fully data-driven end-to-end DST model suitable for verifying our transfer learning algorithm. Figure 2 shows an abbreviated network architecture for the SLU and update modules.

4.1 SLU Module

The SLU module uses word vectors of a user utterance and the feature vector of the system response. Then the module extracts slot values in the form of a dialogue state. The system response consists of a system action tag and slot-value pairs (e.g, $welcomemsg(), confirm(food = italian)$). The system action tag is converted to a one-hot vector $\mathbf{r}_{\text{act}}$ whose dimension is the number of system actions. Slot-value pairs are converted to binary values r_{slot} and r_{val}, where r_{slot} represents whether the system response includes the slot ($r_{\text{slot}} = 1$) or not ($r_{\text{slot}} = 0$) and r_{val} represents whether the system response includes any values ($r_{\text{val}} = 1$) or not ($r_{\text{val}} = 0$). The concatenation of these features is used as the system feature vector $\mathbf{r}$ ($\mathbf{r} = \mathbf{r}_{\text{act}} \oplus r_{\text{slot}} \oplus r_{\text{val}}$ where $\oplus$ is vector concatenation).

Let T be the number of words in the user utterance. Word vectors of the user utterance ($\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_T$) are converted into hidden state vectors ($\mathbf{h}^{\text{f}}_1, \mathbf{h}^{\text{f}}_2, \ldots, \mathbf{h}^{\text{f}}_T, \mathbf{h}^{\text{b}}_1, \mathbf{h}^{\text{b}}_2, \ldots, \mathbf{h}^{\text{b}}_T$) using a bidirectional LSTM as follows:

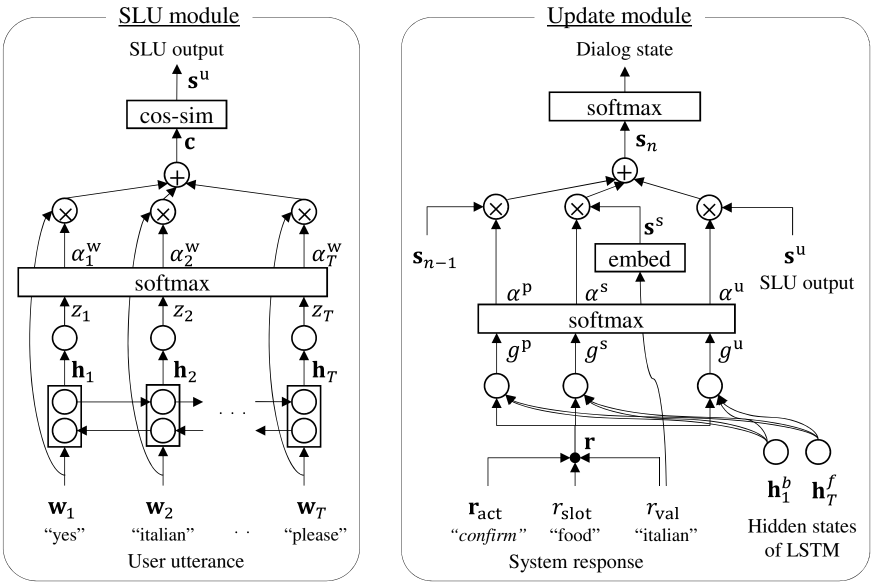


Fig. 2 SLU and update module architecture. Note that the figure for the SLU module focuses on how to encode the user utterance and does not depict how to encode the system response

$$\mathbf{h}_t^{\mathrm{f}} = \mathrm{LSTM}_{\mathrm{fwd}}\left(\mathbf{h}_{t-1}^{\mathrm{f}}, \mathbf{w}_t\right) \ , \tag{2}$$

$$\mathbf{h}_t^{\mathrm{b}} = \mathrm{LSTM}_{\mathrm{bwd}}\left(\mathbf{h}_{t+1}^{\mathrm{b}}, \mathbf{w}_t\right) \ , \tag{3}$$

where $\mathrm{LSTM}_{\mathrm{fwd}}(\cdot, \cdot)$ and $\mathrm{LSTM}_{\mathrm{bwd}}(\cdot, \cdot)$ are forward and backward LSTMs, respectively. Attention weight for each word $(\alpha_1^{\mathrm{w}}, \alpha_2^{\mathrm{w}}, \ldots, \alpha_T^{\mathrm{w}})$ is calculated from the hidden state vectors as follows:

$$z_t = \mathrm{NN}_{\mathrm{usr}}\left(\mathbf{h}_t^{\mathrm{f}} \oplus \mathbf{h}_t^{\mathrm{b}}\right) \ , \tag{4}$$

$$\boldsymbol{\alpha}^{\mathrm{w}} = \mathrm{softmax}\left(z_1 \oplus z_2 \oplus \ldots \oplus z_T\right) \ , \tag{5}$$

where $\boldsymbol{\alpha}^{\mathrm{w}} = \left[\alpha_1^{\mathrm{w}}, \alpha_2^{\mathrm{w}}, \ldots, \alpha_T^{\mathrm{w}}\right]$, and $\mathrm{NN}_{\mathrm{usr}}$ is a one-layer neural network. Attention weights $\boldsymbol{\alpha}^{\mathrm{w}}$ indicate the importance of each word. Context vector $\mathbf{c}$ is calculated from the word vector sequence and attention weights $\boldsymbol{\alpha}^{\mathrm{w}}$ as follows:

$$\mathbf{c} = \sum_{t=1}^{T} \alpha_t^{\mathrm{w}} \mathbf{w}_t \ . \tag{6}$$

Cosine similarity is used to compare the context vector and a word vector of slot values in the ontology. Cosine similarity for the kth value (s_k^{u}) is calculated as follows:

$$s_k^{\mathrm{u}} = \frac{\mathbf{c} \cdot \mathbf{v}_k}{\|\mathbf{c}\| \, \|\mathbf{v}_k\|} \, , \tag{7}$$

where $\mathbf{v}_k$ is a word vector of the kth value, and $\cdot$ is the dot product. Cosine similarity s_k^{u} is calculated for each value.

Next, two biases b^{d} and b^{n} are calculated from the system feature vector $\mathbf{r}$ and the hidden state ($\mathbf{h}_{\mathrm{L}} = \mathbf{h}_T^{\mathrm{f}} \oplus \mathbf{h}_1^{\mathrm{b}}$) as follows:

$$\left[b^{\mathrm{d}}, b^{\mathrm{n}}\right] = \mathrm{NN}_{\mathrm{dn}} \left(\mathbf{h}_{\mathrm{L}} \oplus \mathbf{r}\right) \, , \tag{8}$$

where $\mathrm{NN}_{\mathrm{dn}}$ is a two-layer neural network. Biases b^{d} and b^{n} respectively denote the score for *dontcare* and *None*, where *dontcare* means that the user can accept any values for a slot and *None* means no value is specified. Finally, SLU output $\mathbf{s}^{\mathrm{u}}$ is generated by concatenating the cosine similarity for each value ($s_1^{\mathrm{u}}, s_2^{\mathrm{u}}, \ldots, s_K^{\mathrm{u}}$, where K is the number of slot values) and the two biases b^{d}, b^{n} as follows:

$$\mathbf{s}^{\mathrm{u}} = s_1^{\mathrm{u}} \oplus s_2^{\mathrm{u}} \oplus \ldots \oplus s_K^{\mathrm{u}} \oplus b^{\mathrm{d}} \oplus b^{\mathrm{n}} \, . \tag{9}$$

4.2 Update Module

The update module updates a dialogue state from an SLU output, a raw score for the previous dialogue state, and the system response using attention weights.

First, the similarity between the system response and a value is calculated. From the system response, the module extracts a binary vector ($\mathbf{s}^{\mathrm{s}} = [s_1^{\mathrm{s}}, s_2^{\mathrm{s}}, \ldots, s_K^{\mathrm{s}}]$) whose kth component indicates whether the kth value is included ($s_k^{\mathrm{s}} = 1$) or not ($s_k^{\mathrm{s}} = 0$).

Three attention weights ($\alpha^{\mathrm{p}}, \alpha^{\mathrm{s}}, \alpha^{\mathrm{u}}$) are calculated from the system feature vector $\mathbf{r}$ and the hidden state $\mathbf{h}_{\mathrm{L}}$ as follows:

$$\left[g^{\mathrm{p}}, g^{\mathrm{s}}, g^{\mathrm{u}}\right] = \mathrm{NN}_{\mathrm{att}} \left(\mathbf{h}_{\mathrm{L}} \oplus \mathbf{r}\right) \, , \tag{10}$$

$$\boldsymbol{\alpha} = \mathrm{softmax} \left(g^{\mathrm{p}} \oplus g^{\mathrm{s}} \oplus g^{\mathrm{u}}\right) \, , \tag{11}$$

where $\boldsymbol{\alpha} = [\alpha^{\mathrm{p}}, \alpha^{\mathrm{s}}, \alpha^{\mathrm{u}}]$ and $\mathrm{NN}_{\mathrm{att}}$ is a two-layer neural network. The three attention weights indicate the importance of the raw score of the previous turn, the system response, and the SLU output, respectively.

At turn n, the module calculates a raw score for a dialogue state $\mathbf{s}_n$ by integrating the SLU score $\mathbf{s}^{\mathrm{u}}$, the previous raw score $\mathbf{s}_{n-1}$, and the binary vector of the system response $\mathbf{s}^{\mathrm{s}}$ by attention weights $\boldsymbol{\alpha}$. Finally, the dialogue state $\mathbf{y}_n$ is generated as follows:

$$\mathbf{s}_n = \alpha^{\mathrm{p}} \mathbf{s}_{n-1} + \alpha^{\mathrm{s}} \mathbf{s}^{\mathrm{s}} + \alpha^{\mathrm{u}} \mathbf{s}^{\mathrm{u}} \, , \tag{12}$$

$$\mathbf{y}_n = \mathrm{softmax}(\mathbf{s}_n) \, . \tag{13}$$

5 Experiments

5.1 Experimental Setup

We evaluated the proposed algorithm using the Dialogue State Tracking Challenge 2 (DSTC2) dataset [23] which comprises 1,612 dialogues for training, 506 for validation, and 1,117 for testing. We used three slots, "area", "food" and "pricerange." The "name" slot was excluded because word vectors for several values were not obtained. We used accuracy for each slot as the metric for evaluation. We ran training five times for each slot, and averaged all accuracies.

We implemented the DST model using the neural network framework Chainer [24]. One-best ASR results were used as inputs to the encoding layer described in Sect. 4.1. Contractions were converted to their original forms (e.g, "i'm" to "i am"), then each word was converted to a 300-dimensional word vector using the GloVe model [25], available from the GloVe website.[1] The cell size of the bidirectional LSTM was 32, and the cells were initialized to 0 before training. The cell size of the hidden layers in the two-layer NNs (NN_{dn}, NN_{att}) was 32, and leaky ReLU was used as an activation function for the hidden layers.

For training the model, we used Adam with a learning rate of 0.001, gradient clipping of 1.0, mini-batch size of 32, and 30% dropout for the bidirectional LSTM and NNs. We used word dropout [26] by randomly replacing a word vector with a zero vector for attention weight calculation. The word dropout ratio was set to 40%. Each model was trained with 200 epochs and the best parameter was selected based on the accuracy for the validation data.

5.2 Results

5.2.1 Baseline Performance

Table 1 shows performance of the DST model (Extended attention) described in Sect. 4 without transfer learning (NoTrans). We also show two comparative results for the DSTC2 participants: "Focus baseline," which uses rules [23], and "Delexicalised RNN," which is the DST proposed in [6]. Results for the methods were extracted from the dstc2_results provided on the website.[2] This table shows that our model achieved lower performance than did the others, especially in the food slot. Cause analysis for this result revealed that our model tends to over-retain the previous dialogue state, and then fails to employ the value extracted by the SLU module.

[1] https://nlp.stanford.edu/projects/glove/.

[2] http://camdial.org/~mh521/dstc.

Table 1 Accuracy of each DST model

	Area	Food	Price	Average
Focus baseline	90.8	83.9	92.9	89.2
Delexicalised RNN	**92.4**	**85.6**	**93.0**	**90.3**
Extended attention (NoTrans)	91.0	75.8	91.7	86.1

5.2.2 Comparing Transfer Learning Methods

To validate whether transferring parameters of the update module is effective, we compared methods that transfer parameters of the update module (UpdateTrans), the SLU module (SLUTrans), the bidirectional LSTM of the SLU module (LSTMTrans), and all modules (AllTrans). To measure the performance for unseen slots, we split three slots into two source slots for training transferred parameters, and a target slot for evaluating the DST performance. Note that this comparison did not include sharing the parameters across all existing slots. The transferred parameters were trained from one source slot, and transferred to the target slot. We prepared two settings for the training data sets in the target slot: 1,612 (100%) and 161 (10%). The accuracy measured by each pair of the source and target slots was averaged. We also evaluated the difference in performance when the transferred parameters were fixed or not fixed during training for the target slot.

The "Individual" column in Table 2 shows the performance of each transfer learning method under the described experimental settings. These results show that all methods except for those where transferred parameters are fixed achieve almost the same DST performance in the 100% training data set. This indicates that the 100% training data set is a sufficient amount to train our DST model. As a result, the effectiveness of transfer learning has disappeared. On the other hand, UpdateTrans improved performance over the baseline methods in the 10% training data set, while other comparative methods failed to improve performance. This reveals that transferring parameters of the update module is effective to train a DST model for new slots with a small set of training data. This also suggests that the SLU module and the LSTM of the SLU module have slot-dependent parameters, and that transferring these parameters is insufficient for the target DST model. This suggestion is also derived from the fact that UpdateTrans (Fix parameters = On) achieved higher performance than did others that transferred parameters are fixed in both amounts of training data sets. However, the performance of UpdateTrans (Fix parameters = On) is lower than that of UpdateTrans (Fix parameters = Off). This implies that the update module holds not only slot-independent but also slot-dependent parameters.

Table 2 Accuracy of each transfer learning method. "Individual" transfers parameters trained from one source slot. "Shared" transfers shared parameters trained from all source slots. Experimental settings whose scores are underlined improve performance by sharing parameters

Method	Fix parameters	Individual		Shared	
		Training data		Training data	
		100%	10%	100%	10%
NoTrans	–	86.1	80.7	–	–
SLUTrans	Off	**86.2**	79.9	**86.2**	<u>80.1</u>
LSTMTrans	Off	86.0	80.4	86.0	<u>81.0</u>
AllTrans	Off	85.5	80.6	<u>**86.2**</u>	<u>80.8</u>
UpdateTrans	Off	86.0	**82.5**	<u>**86.2**</u>	**82.5**
SLUTrans	On	65.8	65.7	<u>66.2</u>	<u>66.0</u>
LSTMTrans	On	79.7	74.2	<u>80.8</u>	<u>75.5</u>
UpdateTrans	On	85.1	**81.6**	<u>85.5</u>	<u>**82.0**</u>

5.2.3 Evaluation for Sharing Parameters

We investigated the effectiveness of training shared parameters for transfer across all existing slots. Experimental settings were nearly the same as those described in Sect. 5.2.2; the only difference was that parameters for transfer were trained while the parameters were shared across all source slots.

The "Shared" column in Table 2 shows the performance of each transfer learning method under this settings. UpdateTrans achieved higher performance than did others in the 10% training data set, which is the same trend seen in the results in Sect. 5.2.2. In comparison with training parameters from a single source slot, sharing parameters across all source slots achieved the same or slightly better performance in all transfer learning methods. This indicates that more slot-independent and reliable parameters are obtained by training shared parameters across slots. In contrast, UpdateTrans (Fix parameters = On) still achieved lower performance than did UpdateTrans (Fix parameters = Off). This suggests that the update module still retains slot-dependent parameters. One of possible causes is that the number of source slots is very small. In that case, performance will be improved by increasing the number of source slots.

5.3 *Robustness of Update Module*

We also validated that the proposed transfer learning algorithm did not depend on the network architecture of the update modules. As another module for verification, an update module of a fully statistical neural belief tracking model [11] was used. The update module in this model integrates the SLU output and the dialogue state of the previous turn, then outputs the dialogue state of the current turn via a one-layer neural network. Parameters of the neural network are tied across all slot values, ensuring that the module can deal with unseen values in training. We made the parameters of the neural network tied across all slots, which enabled us to apply the proposed transfer learning. We combined this update module and the SLU module described in Sect. 4.1, and evaluated the DST performance under the experimental settings described in Sect. 5.2.3.

The results show that the DST performance of the proposed methods is higher than that of the baseline methods. Specifically, the performance improved from 57.8% (NoTrans) to 71.2% (UpdateTrans, Fix parameters = Off) in the 10% training data set. It appears that the incompatibility between the SLU module and the update module is one reason why the performance is lower than that of the DST model, which uses the attention-based update module described in Sect. 4.2. In any case, we verified that the proposed methods are effective for another update module, indicating that they are robust to the network architecture of the update modules.

6 Conclusions

This paper proposed a transfer learning algorithm for end-to-end DST to handle unseen slots with a small set of training data. To effectively achieve transfer learning, we assumed that the update module, which is one component of end-to-end DST, can be slot-independent. For obtaining slot-independent parameters of the update module, a DST model for each existing slot was trained while sharing parameters of the update module across all existing slots. Then the slot-independent parameters were transferred to a DST model for a new slot. Evaluation results showed that the proposed algorithm outperformed baseline algorithms, and also suggested that our algorithm was robust to the network architecture of update modules. The advantages of our proposed algorithm will be strengthened by verifying the performance of slots other than those in DSTC2 and the robustness to the network architecture of SLU modules.

References

1. Wang Z, Lemon O (2013) A simple and generic belief tracking mechanism for the dialog state tracking challenge: on the believability of observed information. In: Proceedings of the SIGDIAL 2013 conference, pp 423–432
2. Williams JD (2014) Web-style ranking and slu combination for dialog state tracking. In: Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL), pp 282–291
3. Sun K, Chen L, Zhu S, Yu K (2014) A generalized rule based tracker for dialogue state tracking. In: 2014 IEEE spoken language technology workshop (SLT). IEEE, pp 330–335

4. Perez J, Liu F (2017) Dialog state tracking, a machine reading approach using memory network. In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics, vol 1, pp 305–314
5. Rastogi A, Hakkani-Tür D, Heck L (2017) Scalable multi-domain dialogue state tracking. In: 2017 IEEE automatic speech recognition and understanding workshop (ASRU). IEEE, pp 561–568
6. Henderson M, Thomson B, Young S (2014) Word-based dialog state tracking with recurrent neural networks. In: Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL), Philadelphia, PA, U.S.A., June 2014, Association for Computational Linguistics, pp 292–299
7. Henderson M, Thomson B, Young S (2014) Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In: 2014 IEEE spoken language technology workshop (SLT). IEEE, pp 360–365
8. Mrkšić N, Séaghdha DÓ, Thomson B, Gasic M, Su P-H, Vandyke D, Wen T-H, Young S (2015) Multi-domain dialog state tracking using recurrent neural networks. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing, vol 2, pp 794–799
9. Lukas Z, Filip J (2015) Incremental lstm-based dialog state tracker. In: IEEE workshop on automatic speech recognition and understanding (ASRU). IEEE, pp 757–762
10. Yoshida T, Iwata K, Fujimura H, Akamine M (2018) Dialog state tracking for unseen values using an extended attention mechanism. In: Proceedings of the 8th international workshop on spoken dialogue systems (IWSDS)
11. Mrkšić N, Vulić I (2018) Fully statistical neural belief tracking. In: Proceedings of the 56th annual meeting of the association for computational linguistics, vol 2, pp 108–113
12. Xu P, Hu Q (2018) An end-to-end approach for handling unknown slot values in dialogue state tracking. In: Proceedings of the 56th annual meeting of the association for computational linguistics
13. Zhong V, Xiong C, Socher R (2018) Global-locally self-attentive dialogue state tracker. In: Proceedings of the 56th annual meeting of the association for computational linguistics
14. Ramadan O, Budzianowski P, Gašić M (2018) Large-scale multi-domain belief tracking with knowledge sharing. In: Proceedings of the 56th annual meeting of the association for computational linguistics
15. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
16. Dong W, Fang ZT (2015) Transfer learning for speech and language processing. In: 2015 Asia-Pacific Signal and information processing association annual summit and conference (APSIPA). IEEE, pp. 1225–1237
17. Passban P, Liu Q, Way A (2017) Translating low-resource languages by vocabulary adaptation from close counterparts. ACM Trans Asian Low-Resour Lang Inf Process (TALLIP) 16(4):29
18. Yu J, Qiu M, Jiang J, Huang J, Song S, Chu W, Chen H (2018) Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In: Proceedings of the 11th ACM international conference on web search and data mining. ACM, pp 682–690
19. Lee JY, Dernoncourt F, Szolovits P (2018) Transfer learning for named-entity recognition with neural networks. In: 11th international conference on language resources and evaluation (LREC)
20. Jeong M, Lee GG (2009) Multi-domain spoken language understanding with transfer learning. Speech Commun 51(5):412–424
21. Kim Y-B, Stratos K, Sarikaya R, Jeong M (2015) New transfer learning techniques for disparate label sets. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing, vol 1, pp 473–482
22. Bapna A, Tur G, Hakkani-Tur D, Heck L (2017) Towards zero-shot frame semantic parsing for domain scaling. In: Proceedings of the interspeech, pp 2476–2480

23. Henderson M, Thomson B, Williams J (2014) The second dialog state tracking challenge. In: 15th annual meeting of the special interest group on discourse and dialogue, vol 263
24. Tokui S, Oono K, Hido S, Clayton J (2015) Chainer: a next-generation open source framework for deep learning. In: Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)
25. Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: Empirical methods in natural language processing (EMNLP), pp 1532–1543
26. Iyyer M, Manjunatha V, Boyd-Graber J, Daumé III H (2015) Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing, vol 1, pp 1681–1691

Managing Multi-task Dialogs by Means of a Statistical Dialog Management Technique

David Griol, Zoraida Callejas, and Jose F. Quesada

Abstract One of the most demanding tasks when developing a dialog system consists of deciding the next system response considering the user's actions and the dialog history, which is the fundamental responsibility related to dialog management. A statistical dialog management technique is proposed in this work to reduce the effort and time required to design the dialog manager. This technique allows not only an easy adaptation to new domains, but also to deal with the different subtasks for which the dialog system has been designed. The practical application of the proposed technique to develop a dialog system for a travel-planning domain shows that the use of task-specific dialog models increases the quality and number of successful interactions with the system in comparison with developing a single dialog model for the complete domain.

Keywords Spoken Dialog Systems · Conversational Interfaces · Dialog Management · Domain Knowledge Acquisition · Dialog Structure · Statistical Methodologies

1 Introduction

Spoken conversational interfaces [11] are becoming a strong alternative to traditional graphical interfaces which might not be appropriate for all users and/or applications. These systems can be defined as computer programs that receive speech as

D. Griol (✉) · Z. Callejas
Department of Languages and Computer Systems, University of Granada, Granada, Spain
e-mail: dgriol@ugr.es

Z. Callejas
e-mail: zoraida@ugr.es

J. F. Quesada
Department of Computer Science and Artificial Intelligence,
University of Seville, Seville, Spain
e-mail: jquesada@us.es

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_6

input and generate synthesized speech as output, engaging the user in a dialog that aims to be similar to that between humans. Usually, these systems carry out five main tasks: Automatic Speech Recognition (ASR), Spoken Language Understanding (SLU), Dialog Management (DM), Natural Language Generation (NLG), and Text-To-Speech Synthesis (TTS).

Learning statistical approaches to model these tasks has been of growing interest during the last decade [22]. Models of this kind have been widely used for speech recognition and also for language understanding. Even though in the literature there are models for dialog managers that are manually designed, over the last few years, approaches using statistical models to represent the behavior of the dialog manager have also been developed [7, 10, 21].

However, statistical dialog modeling and parameterization are dependent on expert knowledge, and the success of these approaches is dependent on the quality and coverage of the models and data used for training [18]. To address these important problems, it is important to develop statistical dialog management methodologies able to infer the dialog structure, which implies detecting if users have changed the topic or dialog task, and to deal with unseen situations (i.e., situations that may occur during the dialog and that were not considered during training).

Research on data-driven approaches to dialog structure modeling is relatively new and focuses mainly on recognizing a structure of a dialog as it progresses [24]. Dialog segmentation can be then defined as the process of dividing up a dialog by one of several related criteria (speaker's intention, topic flow, coherence structure, cohesive devices, etc.), identifying boundaries where the discourse changes taken into account such as specific criteria. This detection is usually based on combining different kinds of features, such as semantic similarities, inter-sentence similarities, entity repetition, word frequency, prosodic and acoustic characteristics.

In this paper we propose a practical implementation of a recently developed statistical approach for the development of dialog managers [7], which is mainly based on the use of a classification process for the estimation of a statistical model from the sequences of the system and user actions obtained from a set of training data. The paper is specially focused on the use of specialized dialog models learned for each dialog domain and dialog subtask, instead of learning a generic dialog model for the complete dialog system. To do this, the training data is divided into different subsets, each covering a specific dialog objective or subtask. These specific dialog models are selected by the dialog manager once the objective of the dialog has been detected, using the generic dialog model until this condition has been fulfilled.

We have applied the proposed methodology to develop two versions of a dialog system providing travel-planning information in Spanish. The first one uses a generic dialog model and the second one combines specific classifiers learned for each dialog objective. An in-depth comparative assessment of the developed systems has been completed by means of recruited users. The results of the evaluation show that the specific dialog models allow a better selection of the next system responses, thus increasing the number and quality of successful interactions with the system.

The rest of the paper is organized as follows. Section 2 describes existing approaches for the development of dialog managers, paying special attention to statistical approaches. Section 3 describes our proposal for developing statistical dialog managers with specific dialog models. Section 4 shows the practical implementation of our proposal to develop the two systems for the customer support service. In Sect. 5 we discuss the evaluation results obtained by comparing the two developed systems. Finally, in Sect. 6 we present the conclusions and outline guidelines for future work.

2 State of the Art

As described in the previous section, machine learning approaches to dialog management try to reduce the effort and time required by hand-craft dialog management strategies and, at the same time, to facilitate both to develop new dialog managers and to adapt them to deal with new domains [4].

The most widespread methodology for machine-learning of dialog strategies consists of modeling human-computer interaction as an optimization problem using Markov Decision Processes (MDP) and reinforcement methods [9]. The main drawback of this approach is that the large state space of practical spoken dialog systems, makes its direct re-presentation intractable [23]. Partially Observable MDPs (POMDPs) outperform MDP-based dialog strategies since they provide an explicit representation of uncertainty [16]. This enables the dialog manager to avoid and recover from recognition errors by sharing and shifting probability mass between multiple hypotheses of the current dialog state.

Other interesting approaches for statistical dialog management are based on modeling the system by means of Hidden Markov Models [3], stochastic Finite-State Transducers [15], or using Bayesian Networks [12]. Also [8] proposed a different hybrid approach to dialog modeling in which n-best recognition hypotheses are weighted using a mixture of expert knowledge and data-driven measures, using an agenda and an example-based machine translation approach respectively.

In the literature, there are different methodologies for the application of statistical methodologies for discourse segmentation and the construction of dialog models including task/subtask information. Unsupervised clustering and segmentation techniques are used in [2] to identify concepts and subtasks in task-oriented dialogs.

Diverse machine-learning methodologies have been recently proposed for dialog state tracking (DST) [14, 20], a similar task whose objective is to use the system outputs, user's utterances, dialog context and other external information sources to track what has happened in a dialog. Bayesian dynamic networks are used in generative methods to model a dialog [21]. The main drawback of these methods are that additional dependencies and structures must be learned to consider potentially useful features of the dialog history. The parameters for discriminative methods are directly tuned using machine learning and labeled dialog corpus [13]. Recurrent Neural Networks (RNNs) have been recently proposed as to deal with the high dimensional continuous input features involved in sequential models [19].

3 Our Proposed Methodology for Dialog Management

This section summarizes the proposed dialog management technique and the practical implementation proposed in this paper by means of specific classifiers adapted to each dialog subtask.

3.1 Proposed Statistical Methodology

As described in the introduction section, to develop the *Dialog Manager*, we propose the use of specialized dialog models dealing with each one of the subdomains or subtasks for which the dialog system has been designed.

Our proposed technique for statistical dialog modeling represents dialogs as a sequence of pairs (A_i, U_i), where A_i is the output of the system (the system response or turn) at time i, and U_i is the semantic representation of the user turn (the result of the understanding process of the user input) at time i; both expressed in terms of dialog acts [5]. This way, each dialog is represented by:

$$(A_1, U_1), \ldots, (A_i, U_i), \ldots, (A_n, U_n)$$

where A_1 is the greeting turn of the system (e.g. Welcome to the system. How can I help you?), and U_n is the last user turn (i.e., semantic representation of the last user utterance provided by the natural language understanding component in terms of dialog acts).

The lexical, syntactic and semantic information associated with the speaker u's ith turn (U_i) is denoted as c_i^u. This information is usually represented by:

- the words uttered;
- part of speech tags, also called word classes or lexical categories. Common linguistic categories include noun, adjective, and verb, among others;
- predicate-argument structures, used by SLU modules in various contexts to represent relations within a sentence structure.
- named entities: sequences of words that refer to a unique identifier. This identifier may be a proper name (e.g., organization, person or location names), a time identifier (e.g., dates, time expressions or durations), or quantities and numerical expressions (e.g., monetary values, phone numbers).

Our model is based on the one proposed in [1]. In this model, each system response is defined in terms of the subtask to which it contributes and the system dialog act to be performed.

The term A_i^a denotes the system dialog act (i.e., system action) in the ith turn, and ST_i^a denotes the subtask label to which the ith turn contributes. The interpretation process is modeled in two stages. In the first stage, the system dialog act is determined from the information about the user's turn and the previous dialog context, which is modeled by means of the k previous utterances. This process is shown in Eq. (1).

$$A_i^a = \underset{A^a \in \mathcal{A}}{\operatorname{argmax}} P(A^a | ST_i^a, ST_{i-1}^{i-k}, A_{i-1}^{i-k}, c_{i-1}^{i-k}) \tag{1}$$

where c_i^u represents the lexical, syntactic, and semantic information (e.g., words, part of speech tags, predicate-argument structures, and named entities) associated with speaker u's ith turn; ST_{i-1}^{i-k} represents the dialog subtask tags for utterances $i-1$ to $i-k$; and A_{i-1}^{i-k} represents the system dialog act tags for utterances $i-1$ to $i-k$.

In a second stage, the dialog subtask is determined from the lexical information, the dialog act computed according to Eq. (1), and the dialog context, as shown in Eq. (2).

$$ST_i^a = \underset{s^a \in \mathcal{S}}{\operatorname{argmax}} P(s^a | ST_{i-1}^{i-k}, A_{i-1}^{i-k}, c_{i-1}^{i-k}) \tag{2}$$

The prediction of the dialog subtask (ST_i^a) by means of Eq. (2) is carried out by a specific component in the architecture, which we have called the *Task-Dependent Feature Extractor*. This module is connected with the State of the Dialog Management component, which updates the current state of the dialog according to the semantic information provided by the *Natural Language Understanding* module after each user utterance. This information is provided to the Task-Dependent Feature Extractor for the prediction of the dialog subtask. According to this prediction, the *Task-Dependent Feature Extractor* selects the specialized dialog agent that will be used by the dialog manager in the following turn of the dialog. Then, the selected specialized agent employs the corresponding statistical dialog model to select the next action of the dialog system.

In our proposal, we consider static and dynamic features to estimate the conditional distributions shown in Eqs. (1) and (2). Dynamic features include the dialog act and the task/subtask. Static features include the words in each utterance, the dialog acts in each utterance,and predicate-arguments in each utterance. All pieces of information are computed from corpora using n-grams, that is, computing the frequency of the combination of the n previous words, dialog acts, or predicate-arguments in the user turn.

The conditional distributions shown in Eqs. (1) and (2) can be estimated by means of the general technique of choosing the maximum entropy (MaxEnt) distribution that properly estimates the average of each feature in the training data [1]. This can be written as a Gibbs distribution parameterized with weights λ as Eq. (3) shows, where V is the size of the label set, X denotes the distribution of dialog acts or subtasks (DA_i^u or ST_i^u) and ϕ denotes the vector of the described static and dynamic features used for the user turns from $i-1 \cdots i-k$.

$$P(X = st_i | \phi) = \frac{e^{\lambda_{st_i} \cdot \phi}}{\sum_{st=1}^{V} e^{\lambda_{st_i} \cdot \phi}} \tag{3}$$

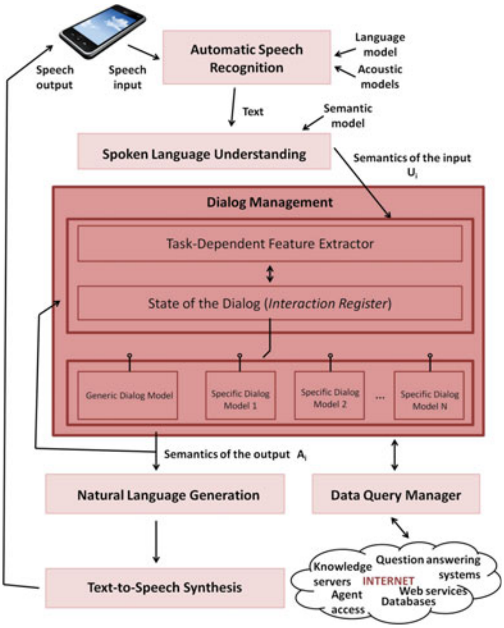


Fig. 1 Scheme of the complete architecture for the development of multitask dialog systems

Such calculation outperforms other state of the art approaches [1], as it increases the speed of training and makes possible to deal with large data sets. Each of the classes can be encoded as a bit vector such that, in the vector corresponding to each class, the ith bit is one and all other bits are zero. Then, V-one-versus-other binary classifiers are used as Eq. (4) shows.

$$P(y|\phi) = 1 - P(\overline{y}|\phi) = \frac{e^{\lambda_y \cdot \phi}}{e^{\lambda_y \cdot \phi} + e^{\lambda_{\overline{y}} \cdot \phi}} = \frac{1}{1 + e^{-\lambda'_{\overline{y}} \cdot \phi}} \quad (4)$$

where $\lambda_{\overline{y}}$ is the parameter vector for the anti-label $\overline{y}$ and $\lambda'_{\overline{y}} = \lambda_y - \lambda_{\overline{y}}$.

Figure 1 shows the described scheme for the practical implementation of the proposed dialog management technique and its interaction with the rest of the modules in the dialog system.

4 Practical Application

We have applied our proposal to develop and evaluate an adaptive system for a travel-planning domain. The system provides context-aware information in natural language in Spanish about approaches to a city, flight schedules, weather forecast, car rental, hotel booking, sightseeing and places of interest for tourists, entertainment guide and theater listings, and movie showtimes. Different Postgress databases are used to store this information and automatically update the data that is included in the application. In addition, several functionalities are related to dynamic information (e.g., weather forecast, flight schedules) directly obtained from webpages and web services providing this information. This way, our system provides a speech access to facilitate this travel-planning information, which is adapted to each user taking into account context information.

Semantic knowledge is modeled in the system using the classical frame representation of the meaning of the utterance. We defined eight concepts to represent the different queries that the user can perform (*City-Approaches*, *Flight-Schedules*, *Weather-Forecast*, *Car-Rental*, and *Hotel-Booking*, *Sightseeing*, *Movie-Showtimes*, and *Theater-Listings*). Three task-independent concepts have also been defined for the task (*Affirmation*, *Negation*, and *Not-Understood*). A total of 101 system actions (DAs) were defined taking into account the information that the system provides, asks or confirms.

Using the *City-Approaches* functionality, it is possible to know how to get to a specific city using the different means of transport. If specific means are not provided by the user, then the system provides the complete information available for the required city. Users can optionally provide an origin city to try to obtain detailed information taking into account this origin. Context information taken into account to adapt this information includes user's current position, and preferred means of transport and city.

The *Flight-Schedules* functionality provides flight information considering the user's requirements. Users can provide the origin and destination cities, ticket class, departure and/or arrival dates, and departure and/or arrival hours. Using the *Weather-Forecast* it is possible to obtain the forecast for the required city and dates (for a maximum of 5 days from the current date). For both functionalities, this information is dynamically extracted from external webpages. Context information taken into account includes user's current location, preferred dates and/or hours, and preferred ticket class.

The *Car-Rental* functionality provides this information taking into account users' requisites including the city, pick-up and drop-off date, car type, name of the company, driver age, and office. The provided information is dynamically extracted from different webpages. The *Hotel-Booking* functionality provides hotels which fulfill the user's requirements (city, name, category, check-in and check-out dates, number of rooms, and number of people).

The *Sightseeing* functionality provides information about places of interest for a specific city, which is directly extracted from the webpage designed for the application. This information is mainly based on users recommendations that have been incorporated in this webpage. The *Theater-Listings* and *Movie-Showtimes* respectively provides information about theater performances and movie showtimes that takes into account the users requirements. These requirements can include the city, name of the theater/cinema, name of the show/movie, category, date, and hour. This information is also considered to adapt both functionalities and then provide context-aware information.

A set of 25 scenarios were manually defined to cover the different queries to perform to the system including different user requirements and profiles. Basic scenarios defined only one objective for the dialog; it means, the user must obtain information about only one type of the possible queries to the system (e.g., to obtain flight schedules from an origin city to a destination for a specific date). More complex scenarios included more than one objective for the dialog (e.g., to obtain information about how to get to a specific city, car rental and hotel booking information).

Two versions of the system have been developed. The first one (*Dialog System 1*) uses a generic dialog model for the task, which employs a single classifier to select the next system response. The second one (*Dialog System 2*) employs 25 specific dialog models, each one of them focused on the achievement of the objective(s) defined for a specific scenario.

5 Results and Discussion

We have completed a comparative evaluation of the two practical dialog systems developed for the task. A total of 150 dialogs were recorded from interactions of 25 users employing the two dialog systems. An objective and subjective evaluation were carried out.

The following measures were defined in the objective evaluation to compare the dialogs acquired with the dialog systems: (i) Dialog success rate; (ii) Dialog length: average number of turns per dialog, number of turns of the shortest dialog, number of turns of the longest dialog, and number of turns of the most observed dialog; (iii) Different dialogs: percentage of different dialogs with respect to the total number of dialogs, and number of repetitions of the most observed dialog; (iv) Turn length: average number of actions per turn; (v) Participant activity: number of turns in the most observed, shortest and longest dialogs; (v) Confirmation rate, computed as the ratio between the number of explicit confirmation turns and the total number of turns in the dialog; and (vi) Error correction rate, computed as the number of errors detected and corrected by the dialog manager divided by the total number of errors.

Table 1 presents the results of the objective evaluation. As can be observed, both dialog systems could interact correctly with the users in most cases for the two systems. However, the *Dialog System 2* obtained a higher success rate, improving the initial results by a 6% absolute. Using the *Dialog System 2*, the average number of required turns is also reduced from 24.3 to 19.1.

It can also be observed that when *Dialog System 2* was used, there was a reduction in the average number of turns and in the number of turns in the longest, shortest and most observed dialogs. These results show that the use of specialized dialog models made it possible to reduce the number of necessary system actions to attain the dialog goals for the different tasks. In addition, the results show a higher variability in the dialogs generated with *Dialog System 2* as there was a higher percentage of different dialogs and the most observed dialog was less repeated. There was also a slight

Table 1 Results of the high-level dialog measures. Dialog success rate (M_1), Average number of turns per dialog (M_2), Percentage of different dialogs (M_3), Repetitions of the most observed dialog (M_4), Average number of actions per turn (M_5), Number of user turns of the most observed dialog (M_6), Number of user turns of the shortest dialog (M_7), Number of user turns of the longest dialog (M_8), Confirmation rate (M_9), Error correction rate (M_{10})

	Dialog System 1	*Dialog System 2*
M_1	89.0%	95.0%
M_2	24.3	19.1
M_3	84.6%	88.7%
M_4	4	3
M_5	1.2	1.5
M_6	12	10
M_7	9	6
M_8	15	11
M_9	38%	36%
M_{10}	0.89%	0.94%

Table 2 Proportions of dialog spent on-goal directed actions, ground actions and other possible actions

	Dialog System 1	*Dialog System 2*
Goal-directed actions	68.21%	74.35%
Grounding actions	30.76%	24.76%
Rest of actions	1.03%	0.89%

increment in the mean values of the turn length for the dialogs collected with *Dialog System 2* due to the better selection of the system actions in the improved strategy.

The confirmation and error correction rates were also improved by using *Dialog System 2* as it required less data from the user, thus reducing the number of errors in the automatic speech recognition process. A problem occurred when the user input was misrecognized but it had high confidence score, in which case it was forwarded to the dialog manager. However, as the success rate shows, this problem did not have a remarkable impact on the performance of the dialog systems.

Additionally, we grouped all user and system actions into three categories: "goal directed" (actions to provide or request information), "grounding" (confirmations and negations), and "other". Table 2 shows a comparison between these categories. As can be observed, the dialogs provided by the *Dialog System 2* have a better quality, as the proportion of goal-directed actions is higher than the values obtained for the *Dialog System 1*.

We also asked the users to complete a questionnaire to assess their subjective opinion about the system performance. The questionnaire had six questions: (i) Q1: *How well did the system understand you?*; (ii)Q2: *How well did you understand the system messages?*; (iii) Q3: *Was it easy for you to get the requested information?*; (iv)

Table 3 Results of the subjective evaluation with recruited users (1 = lowest, 5 = highest)

	Dialog System 1	*Dialog System 2*
Q1	4.7	4.8
Q2	4.3	4.4
Q3	4.2	4.7
Q4	4.2	4.6
Q5	4.1	4.3
Q6	4.4	4.7

Q4: *Was the interaction with the system quick enough?*; (v) Q5: *If there were system errors, was it easy for you to correct them?*; (vi) Q6: *In general, are you satisfied with the performance of the system?* The possible answers for each one of the questions were the same: *Never/Not at all*, *Seldom/In some measure*, *Sometimes/Acceptably*, *Usually/Well*, and *Always/Very Well*. All the answers were assigned a numeric value between one and five (in the same order as they appear in the questionnaire). Table 3 shows the average results of the subjective evaluation using the described questionnaire.

It can be observed that using either *Dialog System 1* or *Dialog System 2* the users perceived that the system understood them correctly. Moreover, they expressed a similar opinion regarding the easiness for correcting system errors. However, users said that it was easier to obtain the information specified for the different objectives using *Dialog System 2*, and that the interaction with the system was more adequate with this dialog manager. Finally, the users were more satisfied with the system employing *Dialog System 2*.

We have completed the evaluation with an additional assessment using a dialog simulation technique [6], which allows to develop a user simulator to automatically interact with a conversational system. The user simulator emulates the user intention, that is, the simulator provides concepts and attributes that represent the intention of the user utterance. Therefore, the user simulator carries out the functions of the ASR and NLU modules. An error simulator module is also integrated to perform error generation and the addition of confidence measures [17].

A total of 3000 dialog were simulated for the two dialog systems developed. The dialogs were only considered successful if they fulfilled the complete list of objectives that had been previously defined for the simulation. Table 4 shows the values obtained for the task success/efficiency measures considered. As it can be observed, the percentage of successfully simulated dialogs increases when the dialog task segmentation module is included. Our analysis also shows that not only the dialogs of the systems including the dialog task segmentation (DTS) module achieve their goals more frequently, but also their average completion time is shorter. The number of different simulated dialogs that are obtained is also increased when this module is included.

Table 4 Results of the high-level dialog features defined for the comparative assessment of the simulated dialogs

	Dialog System 1	*Dialog System 2*
Success. simulated dialogs	83.7%	87.8%
Different dialogs	76.7%	82.9%
Repetit. most seen dialog	11	7
Avg. number of user turns	6.2	5.1

6 Conclusions and Future Work

In this paper, we have described a statistical technique for dialog management in dialog systems. Our proposal is based on dealing with each one of the dialog subtasks or dialog objectives by means of a specific dialog model specialized in each one of them. This model, which considers the previous history of the dialog, is used for the selection of each specialized dialog agent according to the predicted dialog subtask, and the decision of the next system action. Although the construction and parameterization of the dialog model depends on expert knowledge of the task, by means of our proposal, we facilitate to develop dialog systems that have a more robust behavior, better portability, and are easier to be extended or adapted to different user profiles or tasks.

The results of the evaluation of our proposal for a travel-planning dialog system show that the number of successful dialogs is increased in comparison with using a generic dialog agent learned for the complete task. Also, the dialogs acquired using the specific dialog agents are statistically shorter and present a better quality in the selection of the system responses. For future work, we want to consider the incorporation of additional information regarding the user, such as specific user profiles related to their emotional states and adapted to the each application domain.

Acknowledgements The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 823907 (MENHIR project:https://menhir-project.eu) and the CAVIAR project (MINECO, TEC2017-84593-C2-1-R, AEI/FEDER, UE).

References

1. Bangalore S, DiFabbrizio G, Stent A (2008) Learning the structure of task-driven human-human dialogs. IEEE Trans Audio Speech Lang Process 16(7):1249–1259
2. Chotimongkol A (2008) Learning the structure of task-oriented conversations from the corpus of in-domain dialogs. Ph.D. thesis, CMU, Pittsburgh (USA)
3. Cuayáhuitl H, Renals S, Lemon O, Shimodaira H (2005) Human-computer dialogue simulation using hidden Markov models. In: Proceedings of the IEEE workshop on automatic speech recognition and understanding (ASRU'05). San Juan, Puerto Rico, pp 290–295

4. Frampton M, Lemon O (2009) Recent research advances in reinforcement learning in spoken dialogue systems. Knowl Eng Rev 24(4):375–408
5. Griol D, Callejas Z, López-Cózar R, Riccardi G (2014) A domain-independent statistical methodology for dialog management in spoken dialog systems. Comput Speech Lang 28(3):743–768
6. Griol D, Carbó J, Molina J (2013) A statistical simulation technique to develop and evaluate conversational agents. AI Commun 26(4):355–371
7. Griol D, Iglesias J, Ledezma A, Sanchis A (2016) A two-stage combining classifier model for the development of adaptive dialog systems. Int J Neural Syst 26(1)
8. Lee C, Jung S, Kim K, Lee GG (2010) Hybrid approach to robust dialog management using agenda and dialog examples. Comput Speech Lang 24(4):609–631
9. Levin E, Pieraccini R (1997) A stochastic model of human-machine interaction for learning dialog strategies. In: Proceedings of the European conference on speech communications and technology (Eurospeech'97). Rhodes, Greece, pp 1883–1896
10. Lison P (2015) A hybrid approach to dialogue management based on probabilistic rules. Comput Speech Lang 34(1):232–255
11. McTear M, Callejas Z, Griol D (2016) The conversational interface. Springer, Berlin
12. Meng HH, Wai C, Pieraccini R (2003) The use of belief networks for mixed-initiative dialog modeling. IEEE Trans Speech Audio Process 11(6):757–773
13. Metallinou A, Bohus D, Williams J (2013) Discriminative state tracking for spoken dialog systems. In: Proceedings of ACL'13. Sofia, Bulgaria, pp 466–475
14. Pérez J, Liu F (2017) Dialog state tracking, a machine reading approach using Memory Network. In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics (EACL). Valencia, Spain, pp 305–314
15. Planells J, Hurtado L, Sanchis E, Segarra E (2012) An online generated transducer to increase dialog manager coverage. In: Proceedings of the international conference on spoken language processing (Interspeech'2012). Portland, USA
16. Roy N, Pineau J, Thrun S (2000) Spoken dialogue management using probabilistic reasoning. In: Proceedings of the 38th annual meeting of the association for computational linguistics (ACL'00). Hong Kong, China, pp 93–100
17. Schatzmann J, Thomson B, Young S (2007) Error simulation for training statistical dialogue systems. In: Proceedings of IEEE automatic speech recognition and understanding workshop (ASRU'07). Kyoto, Japan, pp 273–282
18. Schatzmann J, Weilhammer K, Stuttle M, Young S (2006) A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. Knowl Eng Rev 21(2):97–126
19. Su P, Mrksic N, Casanueva I, Vulic I (2018) Deep learning for conversational AI. In: Proceeings of the 2018 conference of the North American chapter of the association for computational linguistics (NAACL-HLT 2018). New Orleans, Louisiana, USA, pp 27–32
20. Williams J, Raux A, Henderson M (2016) The dialog state tracking challenge series: a review. Dialogue Discourse 7(3):4–33
21. Williams J, Young S (2007) Partially observable Markov decision processes for spoken dialog systems. Comput Speech Lang 21(2):393–422
22. Young S (2013) Talking to machines. R Acad Eng Ingenia 54:40–46
23. Young S, Schatzmann J, Weilhammer K, Ye H (2007) The hidden information state approach to dialogue management. In: Proceedings of the 32nd IEEE international conference on acoustics, speech, and signal processing (ICASSP). Honolulu, Haway, USA, pp 149–152
24. Zhou Y, Hu Q, Liu J, Jia Y (2015) Combining heterogeneous deep neural networks with conditional random fields for chinese dialogue act recognition. Neurocomputing 168:408–417

Generating Supportive Utterances for Open-Domain Argumentative Dialogue Systems

Koh Mitsuda, Ryuichiro Higashinaka, Taichi Katayama, and Junji Tomita

Abstract Towards creating an open-domain argumentative dialogue system, preparing a database of structured argumentative knowledge for the system as reported in previous work is difficult because diverse propositions exist in the open-domain setting. In this paper, instead of structured knowledge, we use a simple seq2seq-based model to generate supportive utterances to user utterances in an open-domain discussion. We manually collected 45,000 utterance pairs consisting of a user utterance and supportive utterance and proposed a method to augment the manually collected pairs to cover various discussion topics. The generated supportive utterances were then manually evaluated and the results showed that the proposed model could generate supportive utterances with an accuracy of 0.70, significantly outperforming baselines.

1 Introduction

An argumentative dialogue system is an automated agent that can debate with users by responding to user utterances with supportive or non-supportive utterances [7, 11, 12, 15]. Our goal is to create an open-domain argumentative dialogue system that can argue with users about arbitrary discussion topics decided by users.

K. Mitsuda (✉) · R. Higashinaka · T. Katayama · J. Tomita
NTT Media Intelligence Laboratories, Nippon Telegraph and Telephone Corporation, Yokosuka, Japan
e-mail: mitsuda.ko@lab.ntt.co.jp

R. Higashinaka
e-mail: higashinaka.ryuichiro@lab.ntt.co.jp

T. Katayama
e-mail: taichi.katayama@lab.ntt.co.jp

J. Tomita
e-mail: junji.tomita@lab.ntt.co.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_7

Towards creating an open-domain argumentative dialogue system, preparing the structured knowledge of argumentation for the system is difficult. The system relies on a database that has two types of resources: utterances corresponding to propositions stated by the user, and by the system, in argument and relations such as support or non-support among the propositional utterances. The system tries to understand a user utterance by searching the database for a corresponding utterance and then responds with either a supportive or non-supportive utterances related to the one it searched. In previous works, the structured knowledge is formalized in a hierarchy. Thus, in open-domain discussion, it is difficult to create the database because it needs to cover diverse topics and because it is not yet clear how to determine relations among numerous propositional utterances.

To overcome this issue, we propose taking a seq2seq approach [16] that uses manually collected pairs of a user utterance and system utterance in an open-domain discussion. As the first step, we create a model that generates supportive utterances to a user utterance. The coverage of discussion topics in the manually collected data may not be enough to make the model robust to various topics, so we propose methods for automatically bootstrapping and augmenting the manually collected data by utilizing Web resources. In this paper, we report the data collection and construction of the model for generating supportive utterances.

2 Related Work

Although various argumentation mining methods exist [2, 9], only a few studies have reported on argumentative dialogue systems [7, 11, 12, 15].

Sato et al. [15] and Rakshit et al. [12] proposed an argumentative dialogue system based on sentence retrieval from argumentative Web resources on a specific discussion topic. Higashinaka et al. [7] constructed an argumentative dialogue system based on a graph structure that is manually created for a specific discussion topic. Rach et al. [11] proposed an annotation scheme of argumentative texts for utilizing argumentation mining techniques and proposed an argumentative dialogue system based on argument games [1].

In our approach, we do not prepare rich argumentative resources (such as well-organized argumentative texts or hierarchically structured argumentative knowledge) but rather just pairs of a user utterance and supportive utterance. Thus, it is easy to create the argumentative dialogue system.

3 Data Collection

Here, we describe a task description of the supportive utterance generation and the procedure to collect pairs of a user utterance and supportive utterance.

3.1 Supportive Utterance Generation

Figure 1 shows a task description of supportive utterance generation. The input utterance corresponding to a user utterance is an **argumentative utterance** and the output utterance corresponding to a system utterance is a **supportive utterance**. Note that, in this research, Japanese is used as the target language for the data collection and evaluation.

Argumentative utterances describe someone's opinion about a specific discussion topic, which is "cooking" in Fig. 1. An argumentative utterance is limited to a declarative sentence consisting of a subject and predicate for making the task simple, as propositions in real discussion are syntactically too complex [14]. The subject corresponds to the specific topic of discussion ("cooking" in Fig. 1) and the predicate corresponds to an opinion about the topic. Argumentative utterances include not only positive expressions, such as the one in Fig. 1, but also negative expressions (e.g., "Cooking is difficult") or neutral expressions (e.g., "I run every day").

Supportive utterances describe a supportive opinion to the argumentative utterance as the input. A supportive utterance must be a declarative sentence and have a content, such as a word or phrase, different from but related to the argumentative utterance, along with a supporting reason for it. Utterances that can be used as a response to any argumentative utterance, for example, "I think so," are not considered as supportive utterances.

3.2 Collection Procedure

Figure 2 shows the procedure for collecting pairs of an argumentative utterance and supportive utterance. In order to collect pairs that are both varied and general, we used a crowdsourcing platform provided by NTTCom Online Marketing Solutions Corporation. We collected three pairs of an argumentative utterance and supportive utterance from each worker. To simplify the collection procedure and suppress improper working results, we split the procedure of each crowd worker into three steps, listing nouns, writing argumentative utterances, and writing supportive utterances as described below.

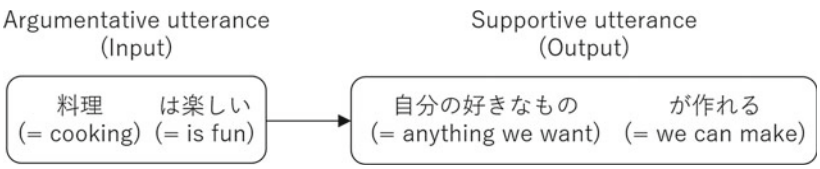


Fig. 1 Supportive utterance generation

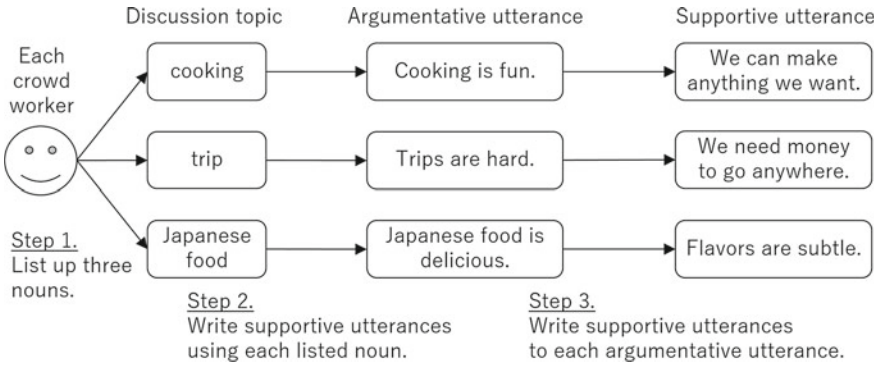


Fig. 2 Data collection of pairs consisting of an argumentative utterance and supportive utterance

Step 1. Listing nouns In this step, crowd workers are instructed to think of and write down three nouns representing things they like, things they do not like, interests, or anything else they can come up with. They are also instructed to avoid writing named entities such as people's names if possible. This is because we want to prioritize the collection of common nouns.

Step 2. Writing argumentative utterances In this step, the workers are instructed to write utterances intended to begin discussions by using the nouns listed up in step 1. They are also instructed to make the written utterances include argumentative contents that some people will support and others will not. Each utterance must consist of a subject and predicate and have at least five characters in Japanese (roughly two or three words in English).

Step 3. Writing supportive utterances In this step, the workers are instructed to write an utterance that supports the content of each argumentative utterance. The written utterances must have a concrete reason for supporting the argumentative utterances and include words different from but related to the argumentative utterances. To come up with reasons for supportive utterances, the workers are instructed to write the utterances in the format: "(argumentative utterance). I think so, because (supportive utterance)." Each utterance must consist of a subject and predicate, and have at least five characters in Japanese.

3.3 Collected Data

Table 1 shows the statistics of the collected pairs of an argumentative utterance and supportive utterance. The "All" column shows the number of collected pairs in total. The 15,000 workers who moved through steps 1 to 3 were recruited from a crowdsourcing platform, so we collected 45,000 pairs in total. We divided the collected pairs into training, development, and test sets in the ratio of 43:1:1. Token length of the training set was limited to between 1 and 32, and those of the development and test sets were limited to between 3 and 10 for eliminating exceptionally short or long utterances. Table 1 includes the vocabulary sizes and token lengths of the

Table 1 Statistics of collected pairs of argumentative utterances (input) and supportive utterances (output). Collected data shown in "All" column is divided into training, development, and test sets

Statistics	Training	Dev.	Test	All
No. of pairs (all)	43,000	1,000	1,000	45,000
No. of pairs (filtered)	42,850	317	397	43,564
Vocabulary size (Input)	11,195	484	531	11,251
Vocabulary size (Output)	13,231	651	729	13,303
Vocabulary size (All)	17,363	962	1047	17,449
Avg. token length (Input)	6.28	5.23	5.17	6.26
Avg. token length (Output)	7.27	6.33	6.18	7.25

Table 2 Example of the collected pairs of an argumentative utterance and supportive utterance written by three workers

Topic	Argumentative utterance	Supportive utterance
Walking	Walking is interesting	It is also good for health
Golf	Golf is difficult	It is hard to get a good score
Work	Worthwhile work is challenging	We can feel accomplishment
Comics	I don't have a chance to read comics	We want to read at our leisure
Movies	I haven't gone to the movie theater recently	Nobody goes often
Music	Music is interesting	Melodies make us happy
Trip	I want to trip everywhere	We want to enjoy various foods
Ukulele	I bought a ukulele	We listen to that in Hawaii
Grand son	My grand son is cute	I can't say no to his requests

filtered pairs. We filtered out 1,436 pairs from the collected data and used 43,564 pairs in this research. From the "All" column and "Avg. token length" rows, we can see that the token length of each argumentative utterance and supportive utterance is from six to seven on average, and supportive utterances tend to be longer than argumentative ones.

Table 2 shows examples of the collected pairs. There are various phrases in both the argumentative and supportive utterances. From each utterance in Table 2, we see that there are various ways of supporting argumentative utterances.

4 Generation Models

Figure 3 shows the proposed method for generating supportive utterances from an argumentative utterance as an input. We introduce two approaches–bootstrapping and data augmentation of the manually collected data–to make the generation model robust to various argumentative utterances. The manually collected data are bootstrapped and augmented before training the generation model.

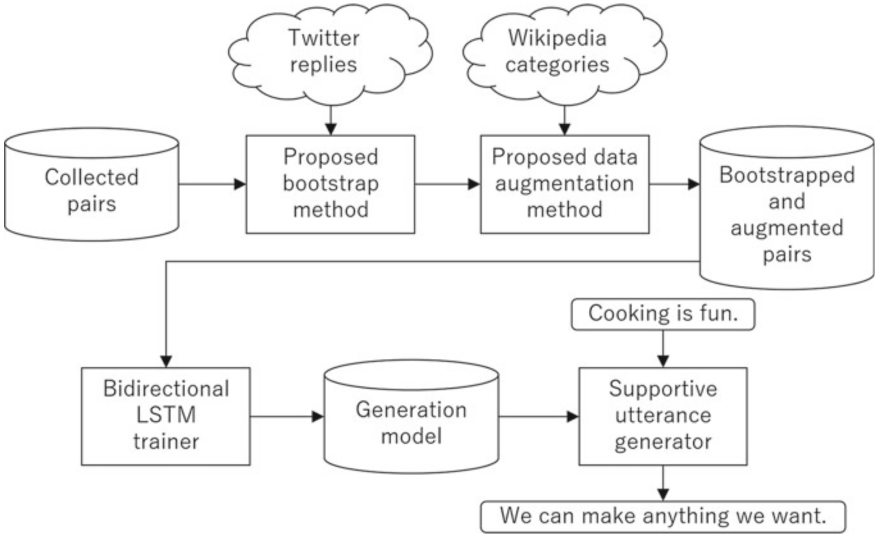


Fig. 3 Proposed method for generating supportive utterances by bootstrapping and augmenting the collected pairs of an argumentative utterance and supportive utterance

4.1 Bootstrapping of Manually Collected Utterance Pairs

Figure 4 shows the flow of bootstrapping the manually collected pairs. We use Twitter replies for extracting pairs similar to the manually collected pairs of an argumentative utterance and supportive utterance. The process consists of keyword listing and Twitter reply filtering by using the keywords.

First, using the manually collected pairs, keywords that tend to appear in the set of argumentative utterances or supportive utterances are listed. For listing the keywords, we use chat utterances collected by Higashinaka et al. [5] as sets of non-argumentative and non-supportive utterances. For listing up the keywords, Pearson's chi-squared test is used for extracting a word that significantly appears in argumentative or supportive utterances.

Table 3 shows the contingency table used for the chi-squared test. The set of chat utterances is compared with the set of the argumentative or the set of the supportive utterances. The values A, B, C, D, and N show the frequency of a target word in each set of utterances. N is a total number of words in the set of chat utterances and the set of the argumentative utterances or the set of the supportive utterances. Using the table, a word is assumed to be a keyword if it significantly appears in the set of the argumentative or supportive utterances.

Twitter reply pairs crawled over six months (from the January 1 to June 13, 2016), including 420,839,296 reply pairs in total, were used for the bootstrap. We empirically set the significance level for keyword listing to 0.01 and the thresholds for reply filtering to 0.5 and 0.6 for argumentative and supportive utterances, respectively. Mention marks, hashtags, and URLs were removed from original tweets, and only

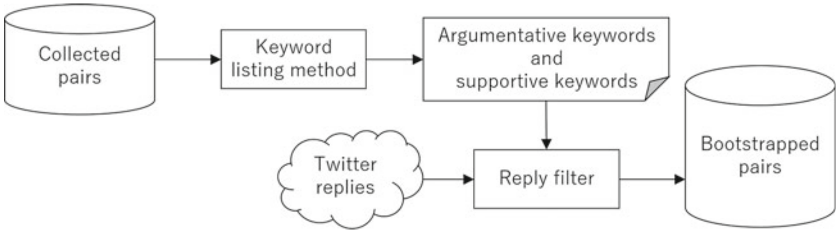


Fig. 4 Bootstrap process of the manually collected pairs consisting of an argumentative utterance and supportive utterance by using Twitter replies

Table 3 Contingency table to classify target words that significantly appear in argumentative utterances or supportive utterances

Set of utterances\Word	Target	Other	Total
Argumentative (or supportive) utterances	A	B	$A + B$
Chat utterances	C	D	$C + D$
Total	$A + C$	$B + D$	N

Table 4 Example of bootstrapped Twitter reply pairs. Words in bold are argumentative keywords or supportive keywords. Bold ratios indicate that a corresponding tweet has keywords over the threshold for extraction. In the "Extracted" column, "1" and "0" correspond to true and false

First tweet		Second tweet		Extracted
Contents	Ratio	Contents	Ratio	
Hello	0% (0/1)	Thank **you** hello	33% (1/3)	0
Why oh why **?**	25% (1/4)	**Because laugh**	**100%** (2/2)	0
Raw vegetables	**50%** (1/2)	**Let's** boil **well**	**66%** (2/3)	1
Spinach **is good for** zinc	**60%** (3/5)	**Vitamin is also abundant**	**100%** (4/4)	1
Instagram **is also good**	**75%** (3/4)	Instagram **seems interesting**	**66%** (2/3)	1

tweets including three to 32 tokens were used. A Japanese morphological analyzer, JTAG [4], was used for this process. As a result of the extraction, we obtained 208,721 Twitter reply pairs, which are 0.05% of the source Twitter pairs and 485% of the manually collected pairs.

Table 4 shows examples of bootstrapped Twitter reply pairs. The replies shown in the upper two rows are the pairs that should not be extracted. These replies barely contain any argumentative or supportive keywords, and thus are not extracted. The replies shown in the lower three rows should be extracted because their contents are both argumentative and supportive. These replies have the keywords in each tweet, and thus are extracted as the pairs of an argumentative utterance and supportive utterance. There are new discussion topics, such as "raw vegetables," "spinach," and "Instagram" in the extracted pairs. The model trained by the bootstrapped pairs can cover more topics than the model trained only by manually collected data.

4.2 Data Augmentation of Argumentative Utterances

In generating supportive utterances, there is a problem when a discussion topic described as a noun is an unknown word (<unk>) or has sparse training data. As a conventional method for solving this problem, pre-trained embeddings are used. However, it is not assumed that similarity in the pre-trained embeddings is the same as similarity in the supportive utterances. In analyzing the manually collected pairs, we assume that discussion topics categorized in the same category, e.g., dogs and cats categorized in the same category of pets, tend to be supported by similar supportive utterances. Thus, we expect that a generation model where topics in similar categories are treated as similar information will be robust to unknown or sparse discussion topics.

In the proposed data augmentation method, we use Wikipedia for augmenting an input noun with categories in Wikipedia. We used category trees, which include a path from each entry to the top-level in Wikipedia [6]. Using Wikipedia, several categories are added to a noun phrase in the first position of an argumentative utterance in preparing the training data or generating supportive utterances. For instance, "surfing" has the path "major category - society - leisure - water sports" and "wakeboarding" has the path "major category - society - leisure - water sports." Using the category information, we can assume that surfing and wakeboarding are similar in terms of the characteristics of leisure and water sports though wakeboarding does not appear in the manually collected data. We add up to two categories for each noun; thus, "surfing is interesting" and "wakeboarding is interesting" are augmented to "leisure water-sports surfing is interesting" and "leisure water-sports wakeboarding is interesting."

Table 5 shows the statistics of the manually collected pairs, bootstrapped pairs, and augmented pairs including the manually collected and bootstrapped ones. As shown in the "Augmented" and "Original" columns, the number of pairs and vocabulary becomes rich. The proposed generation model of supportive utterances are trained by using the data shown in the "Augmented" column.

Table 5 Statistics of manually collected pairs, bootstrapped ones, and data augmented ones. The "Original" column is the same as "Train" in Table 1. "Bootstrapped" shows the pairs only after the bootstrapping. "Augmented" shows all pairs including original pairs and bootstrapped ones after the bootstrapping and augmentation

Statistics	Original	Bootstrapped	Augmented
No. of pairs (all)	43,000	–	–
No. of pairs (filtered)	42,850	208,721	251,571
Vocabulary size (Input)	11,195	36,872	41,448
Vocabulary size (Output)	13,231	28,926	33,768
Vocabulary size (All)	17,363	47,968	53,794
Avg. token length (Input)	6.28	4.84	5.26
Avg. token length (Output)	7.27	4.82	5.28

5 Evaluation

Using the manually collected pairs, we trained the generation model and evaluated the supportive utterances automatically generated to argumentative utterances in the test set shown in Table 1.

5.1 Types of Models

We prepared seven models consisting of four baseline models and three proposed models. One model is not seq2seq-based and the other six models are seq2seq-based. Differences among the following sea2seq-based models are only the training data.

Baseline-retrieval This is the only model that does not generate responses, but search responses from the training data with BM25. It searches the most similar argumentative utterance in the training data and responds with a supportive utterance paired to the searched argumentative utterance.

Baseline-normal This model is a seq2seq-based generation model. It is trained by the training data in the manually collected pairs shown in Table 1 over ten epochs and the best one used for evaluation is determined by a perplexity in the development data. OpenNMT [8] is used for training. The encoder is a bidirectional LSTM, the vocabulary size is 50,000 words, and the source vocabulary and target vocabulary are common. These parameters are the same as those in the other seq2seq-based models.

Baseline-twitter This model has a pre-training process that uses 500,000 randomly extracted Twitter reply pairs and is fine-tuned by using the manually collected pairs. This model is a baseline compared with the Proposed-bootstrap model, which is trained by the filtered Twitter reply pairs.

Baseline-word2vec This model uses pre-trained word embeddings for encoding. The embeddings is trained by one year's worth of Japanese blog texts with word2vec [10, 13]. This model is a baseline compared with the Proposed-augmentation model, which uses only categorical similarities of discussion topics in Wikipedia.

Proposed-bootstrap This model is pre-trained only by the bootstrapped pairs and is fine-tuned by the manually collected pairs.

Proposed-augmentation This model is trained only by the manually collected data that is augmented by Wikipedia.

Proposed-all This model is pre-trained by the bootstrapped and augmented pairs and fine-tuned by the manually collected pairs.

Table 6 Evaluation of generated supportive utterances from each model. The "Accuracy" column shows the ratios of the outputs evaluated as supportive on average between two annotators. The "Statistical test" column shows the results of a Steel-Dwass multiple comparison test [3]. ">" means a model is statistically better ($p < .05$) than a model shown by a corresponding model number (No.)

No.	Model	Accuracy	Statistical test
1	Baseline-retrieval	0.56 (223 / 397)	–
2	Baseline-normal	0.63 (251 / 397)	–
3	Baseline-word2vec	0.62 (247 / 397)	–
4	Baseline-twitter	0.61 (242 / 397)	–
5	Proposed-augmentation	0.65 (260 / 397)	> No. 1
6	Proposed-bootstrap	0.64 (253 / 397)	> No. 1
7	Proposed-all	**0.70** (**279** / 397)	> No. 1, No. 2, No. 3, No. 4

5.2 Evaluation Results

For each argumentative utterance in the test set, seven models generated output utterances as supportive utterances for evaluation. The top output utterances from each model were used for evaluation. Two expert annotators evaluated whether each utterance was supportive or not. An output utterance where no ungrammatical words were included and words were related to the input but semantically different from the input was assumed to be supportive and evaluated as correct, while the others were evaluated as incorrect.

Table 6 shows the results of the evaluation with the average accuracy between two annotators. The results show that the **Proposed-all** model, which is trained by our proposed methods, had the highest accuracy. We statistically tested the accuracy among the models by using a Steel-Dwass multiple comparison test [3]. The results show that **Proposed-all** model was only statistically better ($p < .05$) than the **Baseline-normal** model. This demonstrates that combining the proposed bootstrapping and data augmentation is effective for generating supportive utterances. Although the **Proposed-bootstrap** model and **Proposed-augmentation** model had a tendency to outperform the compared models, namely, the **Baseline-twitter** model and **Baseline-word2vec** model, there was no significant difference among them.

Table 7 shows examples of the top 1 output utterances from each model to the three input utterances in the test set. In the first input, the discussion topic "cooking" frequently appeared in the manually collected data, and each model thus correctly generated supportive utterances.

The second input shown in Table 7, the discussion topic "women announcers" did not appear in the manually collected data, so the **Baseline-normal** model generated incorrect supportive utterances. In our proposed model, the bootstrap method extracted pairs referring to an announcer, for example, a pair of the argumentative utterance "an announcer is nice" and supportive utterance: "everyone is cute," and the data augmentation method converted "women announcer" into "Japanese-announcer

Table 7 Example of supportive utterances generated from each model. The "$Anno_1$" and "$Anno_2$" columns show the annotation results, where "1" and "0" indicate correct and incorrect

Model	Generated supportive utterance	$Anno_1$	$Anno_2$
Input	Cooking is interesting	–	–
Baseline-retrieval	We can find new recipes	0	1
Baseline-normal	We can cook well	0	1
Baseline-word2vec	We can make families happy	1	1
Baseline-twitter	We can eat delicious food	1	1
Proposed-augmentation	We can make anything we like	1	1
Proposed-bootstrap	Families become happy	1	1
Proposed-all	We can make anything we like	1	1
Input	Women announcers are nice	–	–
Baseline-retrieval	Girls like Instagram	0	0
Baseline-normal	We are tired of watching	0	0
Baseline-word2vec	Most of them are handsome guys	0	0
Baseline-twitter	Women are cute	0	0
Proposed-augmentation	They have a dream	1	0
Proposed-bootstrap	Most of them are cute	1	1
Proposed-all	They look pretty	1	1
Input	I want to go to Disney-land in the summer	–	–
Baseline-retrieval	New events will begin soon	1	1
Baseline-normal	Traveling is interesting	0	0
Baseline-word2vec	Traveling is good	0	0
Baseline-twitter	We want to go every year	0	0
Proposed-augmentation	We have new awareness	0	0
Proposed-bootstrap	Traveling is interesting	0	0
Proposed-all	Traveling is interesting	0	0

Japanese-announcer women announcer." Thus, supportive utterances to announcers, who are Japanese women, are additionally trained and the **Proposed-all** model can generate the correct supportive utterance.

The third input shown in Table 7 is a problem that needs to be solved in a future work. As shown in the output of the **Baseline-retrieval** model, the training data contained an argumentative utterance referring to the named entity "Disney-land" but the proposed models could not generate correct supportive utterances referring to things related to Disney-land. We recognize that the number of collected pairs for named entities is not sufficient, and we want to solve this problem by manually collecting data focused on named entities or by creating a method for converting named entities to the well-known common nouns in the training data.

6 Conclusion

This paper presents a model for generating supportive utterances in an open-domain discussion. First, we defined the task of supportive utterance generation from argumentative utterances. Pairs of an argumentative utterance and supportive utterance are manually collected by crowdsourcing. To make the generation model robust to various discussion topics, the collected pairs used for training a seq2seq model are bootstrapped by Twitter replies and augmented by Wikipedia categories. Human annotators evaluated the generated supportive utterances and the results showed that the proposed model generated supportive utterances with the accuracy of 0.70 and statistically outperformed baseline models, including an example-based model (by 0.14) and a model trained only by the manually collected pairs (by 0.07).

For future work, we will construct a model that generating non-supportive utterances for building an argumentative dialogue system. Another interesting future direction is to construct a model that can generate further supportive utterances to the generated supportive utterances from the input argumentative utterance to enable multi-turn discussion. Using such generation models, we plan to investigate strategies for an argumentative dialogue system that can persuade users in the future.

References

1. Bench-Capon TJ (1998) Specification and implementation of toulmin dialogue games. Proc. JURIX 5:5–20
2. Cabrio E, Villata S (2018) Five years of argument mining: a data-driven analysis. In: Proceedings of IJCAL (2018)
3. Dwass M (1960) Some k-sample rank-order tests. Contributions to probability and statistics (1960)
4. Fuchi T, Takagi S (1998) Japanese morphological analyzer using word co-occurrence -JTAG-. In: Proceedings of COLING
5. Higashinaka R, Imamura K, Meguro T, Miyazaki C, Kobayashi N, Sugiyama H, Hirano T, Makino T, Matsuo Y (2014) Towards an open domain conversational system fully based on natural language processing. In: Proceedings of COLING, pp 928–939
6. Higashinaka R, Sadamitsu K, Saito K, Makino T, Matsuo Y (2012) Creating an extended named entity dictionary from wikipedia. In: Proceedings of COLING
7. Higashinaka R, Sakai K, Sugiyama H, Narimatsu H, Arimoto T, Fukutomi T, Matsui K, Ijima Y, Ito H, Araki S, Yoshikawa Y, Ishiguro H, Matsuo Y (2017) Argumentative dialogue system based on argumentation structures. In: Proceedings of SemDial
8. Klein G, Kim Y, Deng Y, Senellart J, Rush AM, OpenNMT: open-source toolkit for neural machine translation
9. Lippi M, Torroni P (2016) Argumentation mining: state of the art and emerging trends. ACM Trans Internet Technol 16(10)
10. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. In: Proceedings of CoRR
11. Rach N, Minker W, Ultes S (2018) Utilizing argument mining techniques for argumentative dialogue systems. In: Proceedings of IWSDS
12. Rakshit G, Bowden KK, Reed L, Misra A, Walker M (2017) Debbie, the debate bot of the future. In: Proceedings of IWSDS (2017)

13. Řehůřek R, Sojka P (2010) Software framework for topic modelling with large corpora. In: Proceedings of LREC workshop on new challenges for NLP frameworks
14. Sakai K, Inago A, Higashinaka R, Yoshikawa Y, Ishiguro H, Tomita J (2018) Creating large-scale argumentation structures for dialogue systems. In: Proceedings of LREC
15. Sato M, Yanai K, Yanase T, Miyoshi T, Iwayama M, Sun Q, Niwa Y (2015) End-to-end argument generation system in debating. In: Proceedings of ACL-IJCNLP
16. Vinyals O, Le Q (2015) A neural conversational model. In: Proceedings of ICML deep learning workshop

VOnDA: A Framework for Ontology-Based Dialogue Management

Bernd Kiefer, Anna Welker, and Christophe Biwer

Abstract We present VOnDA, a framework to implement the dialogue management functionality in dialogue systems. Although domain-independent, VOnDA is tailored towards dialogue systems with a focus on social communication, which implies the need of a long-term memory and high user adaptivity. For these systems, which are used in health environments or elderly care, margin of error is very low and control over the dialogue process is of topmost importance. The same holds for commercial applications, where customer trust is at risk. VOnDA's specification and memory layer relies upon an extended version of RDF/OWL, which provides a universal and uniform representation, and facilitates interoperability with external data sources, e.g., from physical sensors.

1 Introduction

Natural language dialogue systems[1] are becoming more and more popular, be it as virtual assistants such as Siri or Cortana, as Chatbots on websites providing customer support, or as interface in human-robot interactions in areas ranging from human-robot teams in industrial environments [17] over social human-robot-interaction [1] to disaster response [12].

[1]Resource Description Framework https://www.w3.org/RDF/ Web Ontology Language https://www.w3.org/OWL/.

B. Kiefer (✉) · A. Welker · C. Biwer
German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany
e-mail: kiefer@dfki.de; kiefer@semvox.de

A. Welker
e-mail: welanna@web.de

C. Biwer
e-mail: biwer@christophe.lu

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_8

A central component of most systems is the *dialogue manager*, which controls the (possibly multi-modal) reactions based on external triggers and the current internal state. When building dialogue components for robotic applications or in-car assistants, the system needs to take into account inputs in various forms, first and foremost the user utterances, but also other sensor input that may influence the dialogue, such as information from computer vision, gaze detection, or even body and environment sensors for cognitive load estimation.

In the following, we will describe VOnDA, an open-source framework initially developed to implement dialogue strategies for conversational robotic and virtually embodied agents. The implementation mainly took place in the context of the ALIZ-E and PAL projects, where a social robotic assistant supports diabetic children managing their disease. This application domain dictates some requirements that led in to the decision to go for a rule-based system with statistical selection and RDF/OWL underpinning.

Firstly, it requires a lot of control over the decision process, since mistakes by the system are only tolerable in very specific situations, or not at all. Secondly, it is vital to be able to maintain a relationship with the user over a longer time period. This requires a long-term memory which can be efficiently accessed by the dialogue system to exhibit familiarity with the user in various forms, e.g., respecting personal preferences, but also making use of knowledge about conversations or events that were part of interactions in past sessions. For the same reason, the system needs high adaptability to the current user, which means adding a significant number of variables to the state space. This often poses a scalability problem for POMDP-based approaches, both in terms of run-time performance, and of probability estimation, where marginal cases can be dominated by the prominent situation. A third requirement for robotic systems is the ability to process streaming sensor data, or at least use aggregated high-level information from this data in the conversational system.

Furthermore, data collection for user groups in the health care domain is for ethical reasons even more challenging than usual, and OWL reasoning offers a very flexible way to access control.

VOnDA data therefore specifically targets the following design goals to support the system requirements described before:

- Flexible and uniform specification of dialogue semantics, knowledge and data structures
- Scalable, efficient, and easily accessible storage of interaction history and other data, like real-time sensor data, resulting in a large information state
- Readable and compact rule specifications, facilitating access to the underlying RDF database, with the full power of a programming language
- Transparent access to standard programming language constructs (Java classes) for simple integration with the host system

VOnDA is not so much a complete dialogue management system as rather a fundamental implementation layer for creating complex reactive systems, being able to emulate almost all traditional rule- or automata-based frameworks. It provides a strong and tight connection to a reasoning engine and storage, which makes it possible to explore various research directions in the future.

In the next section, we review related work that was done on dialogue frameworks. In Sect. 3, we will give a high-level overview of the VOnDA framework, followed by a specification language synopsis. Section 5 covers some aspects of the system implementation. Section 6 describes the application of the framework in the PAL project's integrated system. The paper concludes with a discussion of the work done, and further directions for research and development.

2 Related Work

The existing frameworks to implement dialogue management components roughly fall into two large groups, those that use symbolic information or automata to specify the dialogue flow (IrisTK [18], RavenClaw [3], Visual SceneMaker [7]), and those that mostly use statistical methods (PyDial [20], Alex [8]). Somewhat in between these is OpenDial [13], which builds on probabilistic rules and a Bayesian Network.

For reasons described in the introduction, VOnDA currently makes only limited use of statistical information. A meaningful comparison to purely learned systems like PyDial or Alex therefore becomes more complex, and would have to be done on an extrinsic basis, which we can not yet provide. We studied comparable systems focusing mainly on two aspects: the specification of behaviours, and the implementation of the dialogue memory / information state.

The dialogue behaviours in IrisTK and SceneMaker are specified using state charts (hierarchical automata). Additional mechanisms (parallel execution, history keeping, exception mechanisms like interruptive edges) make them more flexible and powerful than basic state charts, but their flexibility and generalisation capabilities are limited.

RavenClaw [3] uses so-called *task trees*, a variant of flow charts that can be dynamically changed during run-time to implement dialogue agents for different situations in the dialogue, and an *agenda*, which selects the appropriate agent for the current dialogue state. The resemblance to agent-based architectures using preconstructed plans is striking, but the improved flexibility also comes at the cost of increased complexity during implementation and debugging.

OpenDial [13] tries to combine the advantages of hand-crafted systems with statistical selection, using probabilistic rules which can be viewed as templates for probabilistic graphical models. The parameters for the models can be estimated using previously collected data (supervised learning), or during the interactions with reinforcement learning techniques. Being able to specify structural knowledge for the statistical selection reduces the estimation problem if only a small amount of data is available, and allows to explicitly put restrictions on the selection process.

3 High-Level System Description

VOnDA follows the Information State / Update paradigm [19]. The information state represents everything the dialogue agent knows about the current situation, e.g., containing information about dialogue history, the belief states of the participants, situation data, etc., depending on the concrete system. Any change in the information state will trigger a reasoning mechanism, which may result in more changes in the information state, or outputs to the user or other system components.

VOnDA implements this paradigm by combining a rule-based approach with statistical selection, although in a different way than OpenDial. The rule specifications are close to if-then statements in programming languages, and the information state is realised by an RDF store and reasoner with special capabilities (HFC [10]), namely the possibility to directly use n-tuples instead of triples. This allows to attach temporal information to every data chunk [9, 11]. In this way, the RDF store can represent *dynamic objects*, using either *transaction time* or *valid time* attachments, and as a side effect obtain a complete history of all changes. HFC is very efficient in terms of processing speed and memory footprint, and has recently been extended with stream reasoning facilities. VOnDA can use HFC either directly as a library, or as a remote server, also allowing for more than one database instance, if needed. The initial motivation for using an RDF reasoner was our research interest in multi-session, long-term interactions. In addition, this also allows processing incoming facts in different layers. Firstly, there is the layer of custom reasoning rules, which also comprises streaming reasoning, e.g., for real-time sensor data, and secondly the reactive rule specifications, used mainly for agent-like functionality that handles the behavioural part. This opens new research directions, e.g., underpinning the rule conditions with a probabilistic reasoner (Fig. 1).

The RDF store contains the terminological and the dynamic knowledge: specifications for the data types and their properties, as well as a hierarchy of dialogue acts,

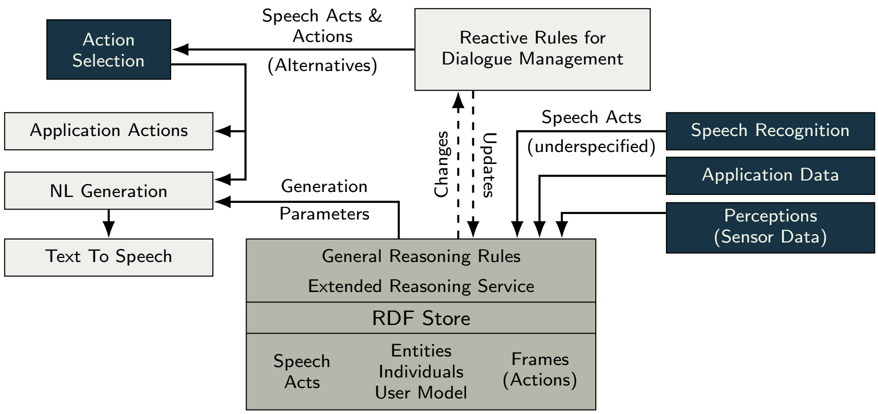


Fig. 1 VOnDA Architecture

semantic frames and their arguments, and the data objects, which are instantiations of the data types. The data type specifications are also used by the compiler to infer the types for property values (see Sect. 4), and form a declarative API to connect new components, e.g., for sensor or application data.

We are currently using the DIT++ dialogue act hierarchy [4] and shallow frame semantics along the lines of FrameNet [16] to interface with the natural language understanding and generation units. Our dialogue act object currently consist of a dialogue act token, a frame and a list of key-value pairs as arguments to the frame (`Offer(Transporting, what=tool, to=workbench)`). While this form of shallow semantics is enough for most applications, we already experience its shortcomings when trying to handle, for example, social talk. Since the underlying run-time core is already working with full-fledged feature matrices, only a small syntax extension will be needed to allow for nested structures.

A set of *reactive condition-action rules* (see Fig. 4) is executed whenever there is a change in the information state. These changes are caused by incoming sensor or application data, intents from the speech recognition, or expired timers. Rules are labelled if-then-else statements, with complex conditions and shortcut logic, as in Java or C. The compiler analyses the base terms and stores their values during processing for dynamic logging. A rule can have direct effects, like changing the information state or executing system calls. Furthermore, it can generate so-called *proposals*, which are (labelled) blocks of code in a frozen state that will not be immediately executed, similar to closures.

All rules are repeatedly applied until a fixed point is reached where no new proposals are generated and there is no information state change in the last iteration. Subsequently, the set of proposals is evaluated by a statistical component, which will select the best alternative. This component can be exchanged to make it as simple or elaborate as necessary, taking into account arbitrary features from the data storage.

A VOnDA project consists of an ontology, a custom extension of the abstract `Agent` class (the so-called *wrapper class*), a client interface to connect the communication channels of the application to the agent, and a set of rule files that are arranged in a tree, using `import` statements. The blue core in Fig. 2 is the run-time system which is part of the VOnDA framework, while all elements above are application specific parts of the agent. A `Yaml` project file contains all necessary information for compilation: the ontology, the wrapper class, the top-level rule file and other parameters, such as custom compile commands.

The ontology contains the definitions of dialogue acts, semantic frames, class and property specifications for the data objects of the application, and other assertional knowledge, such as specifications for "forgetting", which could be modeled in an orthogonal class hierarchy and supported by custom deletion rules in the reasoner.

Every rule file can define variables and functions in VOnDA syntax which are then available to all imported files. The methods from the wrapper class are available to all rule files.

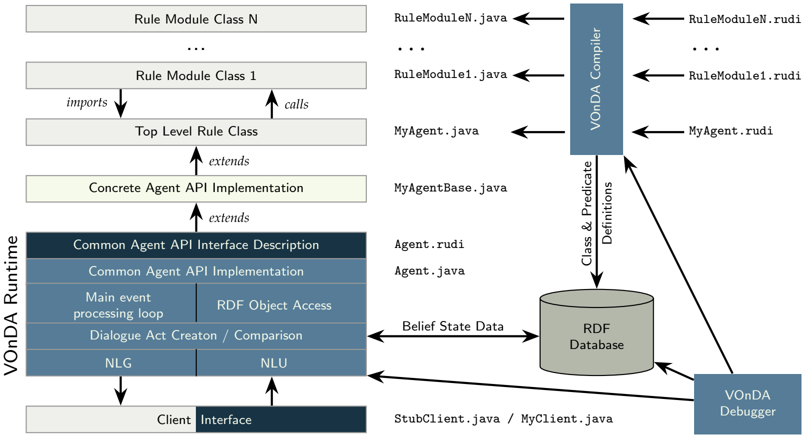


Fig. 2 A schematic VOnDA agent

The current structure assumes that most of the Java functionality that is used inside the rule files will be provided by the `Agent` superclass. There are, however, alternative ways to use other Java classes directly, with support for the same type inference as for RDF classes.

4 Dialogue Specification Language

VOnDA's rule language at first sight looks very similar to Java/C++. However, there are a number of specific features which make it convenient for the implementation of dialogue strategies. Maybe the most important one is the handling of RDF objects and classes, which can be treated similarly to those of object oriented programming languages, including the (multiple) inheritance and type inference that are provided by the RDF class hierarchies.

Figure 3 contains an example of VOnDA code, and how it relates to RDF type and property specifications, schematically drawn on the right. The domain and range definitions of properties are picked up by the compiler and used in various places,

```
user = new Animate;
user.name = "Joe";
set_age:
if (user.age <= 0) {
  user.age = 15;
}
```

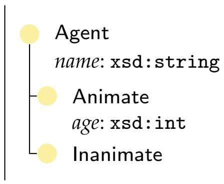


Fig. 3 Ontology and VOnDA code

```
1  if (!saidInSession(#Greeting(Meeting)) {
2    timeout("wait_for_greeting", 7000){ //Wait 7 secs before taking initiative
3      if (! receivedInSession(#Greeting(Meeting))
4        propose("greet") {
5          da = #InitialGreeting(Meeting);
6          if (user.name) da.name = user.name;
7          emitDA(da);
8        }
9    }
10
11   if (receivedInSession(#Greeting(Meeting))
12     propose("greet_back") { // We assume we know the name by now
13       emitDA(#ReturnGreeting(Meeting, name={user.name}));
14     }
15 }
```

Fig. 4 VOnDA code example

e.g., to infer types, do automatic code or data conversions, or create "intelligent" boolean tests, such as the one in line 4, which will expand into two tests, one testing for the existence of the property for the object, and in case that succeeds, a test if the value is greater than zero. If there is a chain of more than one field resp. property access, every part is tested for existence in the target code, keeping the source code as concise as possible. Also, for reasons of brevity, the type of a new variable needs not be given if it can be inferred from the value assigned to it.

New RDF objects can be created with `new`, similar to Java objects; they are immediately reflected in the database, as are all changes to already existing objects.

Many operators are overloaded, especially boolean operators such as **`<=`**, which compares numeric values, but can also be used to test if an object is of a specific class, for subclass tests between two classes, and for subsumption of dialogue acts.

There are two statements with a special syntax and semantics: `propose` and `timeout`. `propose` is VOnDA's current way of implementing probabilistic selection. All (unique) propose blocks that are in active rule actions are collected, frozen in the execution state in which they were encountered, such as closures known from functional programming languages. When rule processing stops, a statistical component picks the "best" proposal and its closure is executed.

`timeouts` also generate closures, but with a different purpose. They can be used to trigger proactive behaviour, or to check the state of the system after some time period, or in regular intervals. A timeout will only be created if there is no active timeout with that name.

Figure 4 also contains an example of the short-hand notation for shallow semantic structures (starting with **#**). Since they predominantly contain constant (string) literals, this is the default when specifying such structures. The special syntax in `user={user.name}` allows to insert the value of expressions into the literal, similar to an *eval*.

This section only described the most important features of VOnDA's syntax. For a detailed description, the reader is referred to the user documentation.[2]

5 Compiler/Run-Time Library

The compiler turns the VOnDA source code into Java source code using the information in the ontology. Every source file becomes a Java class. Although the generated code is not primarily for the human reader, a lot of care has been taken in making it still understandable and debuggable. The compile process is separated into three stages: parsing and abstract syntax tree building, type checking and inference, and code generation.

The VOnDA compiler's internal knowledge about the program structure and the RDF hierarchy takes care of transforming the RDF field accesses into reads from and writes to the database. Beyond that, the type system, resolving the exact Java, RDF or RDF collection type of (arbitrary long) field accesses, automatically performs the necessary casts for the ontology accesses.

The run-time library contains the basic functionality for handling the rule processing, including the proposals and timeouts, and for the on-line inspection of the rule evaluation. There is, however, no blueprint for the main event loop, since that depends heavily on the host application. It also contains methods for the creation and modification of shallow semantic structures, and especially for searching the interaction history for specific utterances. Most of this functionality is available through the abstract `Agent` class, which has to be extended to a concrete class for each application.

There is functionality to directly communicate with the HFC database using queries, in case the object view is not sufficient or too awkward. The natural language understanding and generation components can be exchanged by implementing existing interfaces, and the statistical component is connected by a message exchange protocol. A basic natural language generation engine based on a graph rewriting module is already integrated, and is used in our current system as a template based generator. The example application also contains a VoiceXML based interpretation module.

5.1 Debugger/GUI

VOnDA comes with a GUI [2] that helps navigating, compiling and editing the source files belonging to a project. It uses the project file to collect all the necessary information (Fig. 5).

[2]https://github.com/bkiefer/vonda/blob/master/doc/master.pdf.

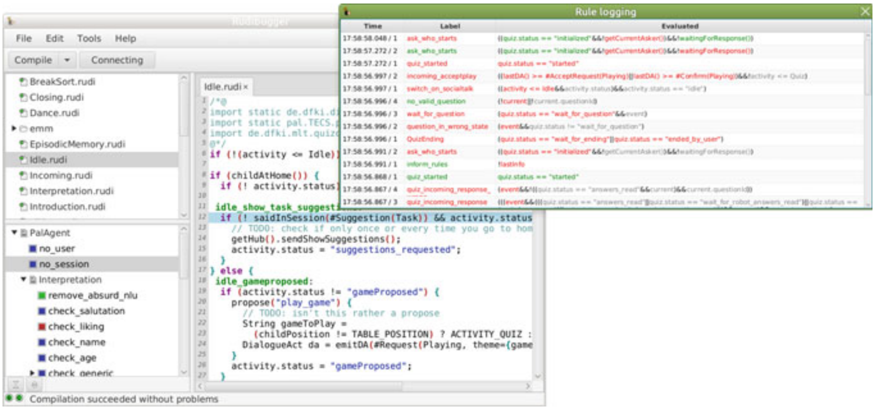


Fig. 5 The VOnDA GUI window

Upon opening a project, the GUI displays the project directory (in a *file view*). The user can edit rule files from within the GUI or with an external editor like Emacs, Vim, etc. and can start the compilation process. After successful compilation, the project view shows what files are currently used, and marks the top-level and the wrapper class files. A second tree view (*rule view*) shows the rule structure in addition to the module structure. Modules in which errors or warnings were reported during compilation are highlighted, and the user can quickly navigate to them using context menus.

Additionally, the GUI can be used to track what is happening in a running system. The connection is established using a socket to allow remote debugging. In the rule view, multi-state check boxes are used to define which rules should be observed under which conditions. A rule can be set to be logged under any circumstances, not at all or if its condition evaluated to true or to false. Since the rules are represented in a tree-like structure, the logging condition can also be set for an entire subgroup of rules, or for a whole module. The current rule logging configuration can be saved for later use.

The *logging view* displays incoming logging information as a sortable table. A table entry contains a time stamp, the rule's label and its condition. The rule's label is coloured according to the final result of the whole boolean expression. Each base term of the condition is coloured accordingly, or greyed out if short-cut logic led to premature failure or success of the expression. Inspecting the live system helps pin-point problems when the behaviour is not as expected. The log shows how the currently active part of the information state is processed, and the window offers easy navigation using the mouse from the rule condition to the corresponding source code.

6 Applications

VOnDA is used in the integrated system of the EU project PAL [15], which uses human-robot interaction to support children with diabetes type 1 in coping with their disease. Children interact with a real NAO robot,[3] or with an Android app that connects to the core system and exhibits a virtual character that is as similar to the robot as possible, also in its behaviour.

The dialogue component, which is largely responsible for the agent's behaviour, is implemented using the VOnDA framework. In addition, HFC, the RDF store that VOnDA builds upon, is the main database of the system, storing all relevant information and being the central data exchange hub. The system runs as a cloud-based robotic solution, spawning a new system instance for every user. It has been successfully tested with more than 40 users at a time on a medium sized virtual machine[4] with only moderate load factors, giving a positive indication of the scalability of HFC and the VOnDA approach.

There are two helper modules integrated into the dialogue component which quite extensively exploit the connection between the database and the rule part, namely the *Episodic Memory* and the *Targeted Feedback*. While the targeted feedback reacts to current events in the running session, like entering a bad or good glucose value, or the current achievement of a task, the episodic memory aggregates data from the past and eventually converts them into so-called episodes that are used for interactions in subsequent sessions. Both are only triggered if relevant changes in the database occur, for example incoming data from the MyPAL app about games or achievements, and serve different conversational purposes, namely showing familiarity with the user and her/his everyday life, versus reacting to current positive or negative incidents.

VOnDA has also been used in a recent project aiming to implement a generalised, ontology-based approach to open-domain talk [21]. The Smoto system uses an additional HFC server running WordNet [6, 14] as semantic database, thereby gaining knowledge about semantic concepts that can be used in the dialogue and to find appropriate reactions on arbitrary user input.

7 Discussion and Further Work

We believe that there are still many interesting application areas for hybrid statistical and hand crafted systems, e.g., if they are relatively small, or there is little domain-specific data available. Many currently deployed systems that build on much simpler technology like VoiceXML can certainly profit from hybrid approaches such as OpenDial or VOnDA.

VOnDA is under active development. We designed it such that it can be integrated in most applications and opens many ways for improvements and additions. As a

[3]Softbank Robotics https://www.ald.softbankrobotics.com.

[4]4 core Xeon E5-2683@2.00GHz, 16 GB RAM.

rule-based framework that is close to being a programming language, VOnDA is able to completely emulate the automata-based frameworks. In fact, we are currently working on a graphical editor à la SceneMaker and the precompilation of hierarchical state charts into VOnDA code. We hope this will facilitate the implementation of new applications for inexperienced users and help with rapid prototyping, while retaining the greater flexibility and modularization capabilities. In this way, we combine the intuitive way of specifying simple strategies with the full flexibility of the framework.

VOnDA could also be used to implement modules that simulate the agents of RavenClaw. To get a functionality similar to RavenClaw's agenda, its action selection module would have to be implemented as a dialogue state tracker, activating the most probable agent at each dialogue step.

Using the well-established RDF/OWL standard as specification layer makes it very easy to add or change application specific data structures, especially because of the existing tool support. We already use the reasoning facilities for type and partially for temporal inference, but given the possibility of attaching also confidence or credibility information to the RDF data, a more integrated probabilistic approach with soft preconditions could be implemented, e.g., on the basis of Dempster-Shafer theory [5]. Moreover, additional meta knowledge, such as trustworthiness or validity periods could be declared using multiple inheritance, which opens many interesting research directions.

Other next steps will be the addition of default adaptors for obviously needed external modules like automatic speech recognition, more flexible language understanding, and the like. We will also work on the improvement of the GUI, including features such as a watch window and/or a timeline to track changes of specific values in the database, and a tool that analyses the dependencies between rules on the basis of the conditions' base terms.

From the research perspective, there are two very interesting lanes: integrating probabilistic reasoning as a first-class option, which is directly integrated with the rule conditions, and adding an additional layer to facilitate the implementation of BDI-like agents, to study the connections and dependencies between conversational and non-conversational behaviours.

7.1 Source Code and Documentation

The VOnDA core system can be downloaded at https://github.com/bkiefer/vonda.git. The main page has detailed instructions for the installation of external dependencies. The debugger currently lives in a separate project: https://github.com/yoshegg/rudibugger.git. Both projects are licensed under the Creative Commons Attribution-NonCommercial 4.0 International License,[5] and are free for all non-commercial use. A screen cast showing the GUI functionality and the running PAL system is available at https://youtu.be/nSotEVZUEyw.

[5] http://creativecommons.org/licenses/by-nc/4.0/.

Acknowledgements The research described in this paper has been funded by the Horizon 2020 Framework Programme of the European Union within the project PAL (Personal Assistant for healthy Lifestyle) under Grant agreement no. 643783.
This paper is dedicated to our colleague and friend Hans-Ulrich Krieger, the creator of HFC. Hope you found peace, wherever you are.

References

1. ALIZ-E project (2010) http://www.aliz-e.org/
2. Biwer C (2017) Rudibugger - Graphisches Debugging der Dialogmanagementtechnolgie VOnDA. Bachelor's Thesis, Saarland University. https://doi.org/10.13140/RG.2.2.36556.31368
3. Bohus D, Rudnicky AI (2009) The RavenClaw dialog management framework: architecture and systems. Comput Speech Lang 23(3):332–361
4. Bunt H, Alexandersson J, Choe JW, Fang AC, Hasida K, Petukhova V, Popescu-Belis A, Traum DR (2012) ISO 24617-2: a semantically-based standard for dialogue annotation. In: LREC. Citeseer, pp 430–437
5. Dempster AP (2008) The Dempster-Shafer calculus for statisticians. Int J Approx Reason 48(2):365–377
6. Fellbaum C (1998) WordNet. Wiley Online Library
7. Gebhard P, Mehlmann G, Kipp M (2012) Visual SceneMaker—a tool for authoring interactive virtualcharacters. J Multimodal User Interfaces 6(1–2):3–11
8. Jurčíček F, Dušek O, Plátek O, Žilka L (2014) Alex: a statistical dialogue systems framework. In: International conference on text, speech, and dialogue. Springer, pp 587–594
9. Krieger HU (2012) A temporal extension of the Hayes/ter Horst entailment rules and an alternative to W3C's n-ary relations. In: Proceedings of the 7th international conference on formal ontology in information systems (FOIS), pp 323–336
10. Krieger HU (2013) An efficient implementation of equivalence relations in OWL via rule and query rewriting. In: 2013 IEEE 7th international conference on semantic computing (ICSC). IEEE, pp 260–263
11. Krieger HU (2014) A detailed comparison of seven approaches for the annotation of time-dependent factual knowledge in rdf and owl. In: Proceedings 10th joint ISO-ACL SIGSEM workshop on interoperable semantic annotation
12. Kruijff-Korbayová I, Colas F, Gianni M, Pirri F, de Greeff J, Hindriks K, Neerincx M, Ögren P, Svoboda T, Worst R (2015) TRADR project: long-term human-robot teaming for robot assisted disaster response. KI-Künstliche Intelligenz 29(2):193–201
13. Lison P, Kennington C (2015) Developing spoken dialogue systems with the OpenDial toolkit. In: SEMDIAL 2015 goDIAL, p 194
14. Miller GA (1995) Wordnet: a lexical database for english. Commun ACM 38(11):39–41
15. PAL: The PAL Project Website (2018) http://www.pal4u.eu/. Last access 26 May 2020
16. Ruppenhofer J, Ellsworth M, Petruck MR, Johnson CR, Scheffczyk J (2016) FrameNet II: extended theory and practice. Institut für Deutsche Sprache, Bibliothek
17. Schwartz T, Zinnikus I, Krieger HU, Bürckert C, Folz J, Kiefer B, Hevesi P, Lüth C, Pirkl G, Spieldenner T, et al (2016) Hybrid teams: flexible collaboration between humans, robots and virtual agents. In: German conference on multiagent system technologies. Springer, pp 131–146
18. Skantze G, Al Moubayed S (2012) IrisTK: a statechart-based toolkit for multi-party face-to-face interaction. In: Proceedings of the 14th ACM international conference on multimodal interaction. ACM, pp 69–76
19. Traum DR, Larsson S (2003) The information state approach to dialogue management. In: Current and new directions in discourse and dialogue. Springer, pp 325–353

20. Ultes S, Barahona LMR, Su PH, Vandyke D, Kim D, Casanueva I, Budzianowski P, Mrkšić N, Wen TH, Gašić M, et al (2017) Pydial: a multi-domain statistical dialogue system toolkit. In: Proceedings of ACL 2017, system demonstrations, pp 73–78
21. Welker A (2017) Wissensbasiertes Dialogmanagement für Social Talk. Bachelor's Thesis, Saarland University. https://doi.org/10.13140/RG.2.2.16423.65441

Human-Robot Interaction

Towards Increasing Naturalness and Flexibility in Human-Robot Dialogue Systems

Graham Wilcock and Kristiina Jokinen

Abstract The chapter discusses some approaches to increasing the naturalness and flexibility of human-robot interaction, with examples from the WikiTalk dialogue system. WikiTalk enables robots to talk fluently about thousands of topics using Wikipedia-based talking. However, there are three challenging areas that need to be addressed to make the system more natural: speech interaction, face recognition, interaction history. We address these challenges and describe more context-aware approaches taking the individual partner into account when generating responses. Finally, we discuss the need for a Wikipedia-based listening capability to enable robots to follow the changing topics in human conversation. This would allow robots to join in the conversation using Wikipedia-based talking to make new topically relevant dialogue contributions.

1 Introduction

The WikiTalk open-domain spoken dialogue system [7] enables robots to talk about thousands of different topics using information from Wikipedia. By using human-written sentences and paragraphs from Wikipedia, the robot is able to talk fluently and at length about the topics. As Wikipedia is an open community-based project with articles revised and edited by a world-wide community of volunteers, the sentences are grammatically correct and the paragraphs are locally coherent, and the information the robot gives is not only up-to-date but also more trustworthy than information from private sources such as newspapers and magazines.

G. Wilcock
CDM Interact, Helsinki, Finland
e-mail: graham.wilcock@cdminteract.com

K. Jokinen (✉)
Artificial Intelligence Research Center, AIST Tokyo Waterfront, Tokyo, Japan
e-mail: kristiina.jokinen@aist.go.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_9

The robot makes smooth topic shifts using the hyperlinks in Wikipedia, so users can navigate smoothly from topic to topic following their own individual interests. The robot talks about the topics selected by the user, but if the information turns out to be uninteresting or if it is not convenient for the user to listen at the time, the robot can be interrupted by touching the sensor on its head. WikiTalk is multilingual, and currently works in English, Finnish and Japanese.

A video[1] of a robot talking fluently about Shakespeare in English and Japanese and switching languages on demand is described by [17]. Although WikiTalk is one of the few existing open-domain dialogue systems and its implementation on Nao robots includes gesturing to make the interaction livelier and more comprehensible to the user, the video reveals some challenges.

- The robot's contributions are unnatural in a spoken dialogue as it is reading out sentences from written texts.
- The user's dialogue contributions are also unnatural, as they only comprise giving commands like *Continue* or requesting topic names like *Julius Caesar*.
- The robot does not recognize returning users. When a user goes away and comes back, the robot should recognize the user and start the dialogue with a reference to where it was when the user left.

2 Approaches to Improving Naturalness and Flexibility

In this section we address the listed challenging areas. We propose that WikiTalk and similar systems can be improved to offer more natural and flexible interaction especially by increasing the focus on the individual user.

2.1 More Natural Speech by the Robot

WikiTalk processes the sentences that it extracts from Wikipedia to make them more suitable for speech, but the robot is reading out written text. To improve naturalness of the presentation, one option is to modify the style of the written text to better fit spoken dialogue. Style transfer of texts to automatically transfer from written to spoken style using deep learning is an active research area [2] but it has not yet reached the same level as style transfer of images. More importantly, deviating from the Wikipedia text risks losing the trustworthiness of the information. We therefore prefer that the robot follows the written text closely, with minimal changes.

To provide more dialogue-like interaction we use interactive story-telling techniques [3, 6] where generation is based on suitable chunks and on feedback from the user, rather than radically changing the style. WikiTalk segments the text into suitably

[1] https://www.youtube.com/watch?v=NkMkImATfYQ.

sized chunks and elicits feedback from the individual user on whether to continue or not. Currently the feedback is verbal but future work will include recognition of multimodal signals (gaze and gesturing).

2.2 More Natural Speech by the Human

To avoid restricting the user to an unnatural speaking style using only commands like *Continue* or topic names like *Julius Caesar*, WikiTalk employs word-spotting techniques. *Tell me about Julius Caesar* or *Julius Caesar would be interesting* or *What about Julius Caesar?* and so on are recognized as the user wanting to hear about Julius Caesar.

However, word-spotting has limitations, for example longer sentences are problematic, and negation or sarcasm are not recognized. So *Oh no! Not Julius Caesar again!* would be misinterpreted as a request to hear about Julius Caesar. In these cases, more sophisticated NLP tools are required. In most cases, however, the users express their intentions in a factual and positive manner.

2.3 Face Recognition

Recent progress in face recognition includes face landmark detection [10]. Face landmarks detected using Openface [1] are shown in Fig. 1 (left part). The landmarks are used to produce face embeddings using a Facenet-type deep convolutional neural network [16]. The face embeddings are then used to classify new face images.

To recognize known individuals, classifiers compare features of new images with features of previously recorded images labelled with names. Face recognition of known persons (Fig. 1 right part) is crucial for long term interaction, for example in

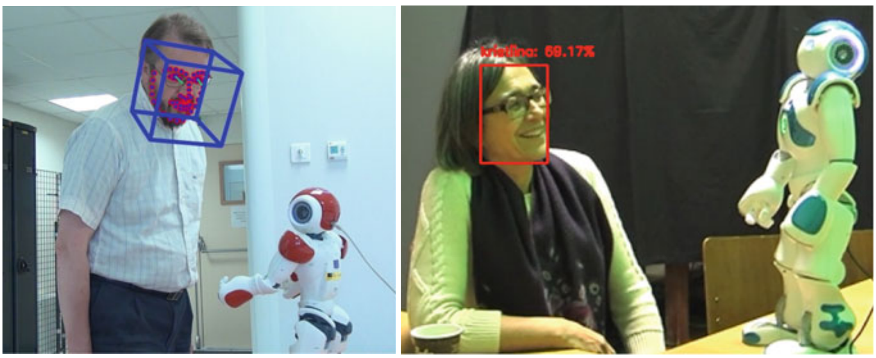


Fig. 1 Face landmarks of the first author and face recognition of the second author

task-oriented dialogues in elder-care homes [5] where robots may give instructions to a specific carer on how to perform a care-giving task for a specific resident.

To recognize recently-seen unknown persons, their new images are compared with recent short-term stored images with anonymous labels such as *unknown-23*. It is thus important that the system also maintains an interaction history.

2.4 User Interaction History

Information about user interactions can be stored short-term or long-term using web apps and cloud data storage. The stored information can include a username or ID, preferred form of address, and preferred language. CDM Interact has used a Google Cloud web app (https://wikitalk-app.cdminteract.com) as shown in Fig. 2. Interaction histories record the Wikipedia topics and chunks presented to the user. This information is needed to enable resuming from previous topics.

The example in Fig. 2 shows that the user previously heard about Kyoto and Kyoto University. These topic histories can be used in backward referencing (*What did we talk about last time?*), summaries (*Previously we talked about Kyoto*), or proactive prediction of topics (changing the probability of *Kyoto* as the next topic).

However, storing any data about user identities and what topics people have shown interest in raises ethical, legal and social issues, which will require appropriate permissions and protections before being used with end users. These issues are important especially for care service applications as discussed in [4], but also for private spaces. We must not allow robots to become spies in our homes.

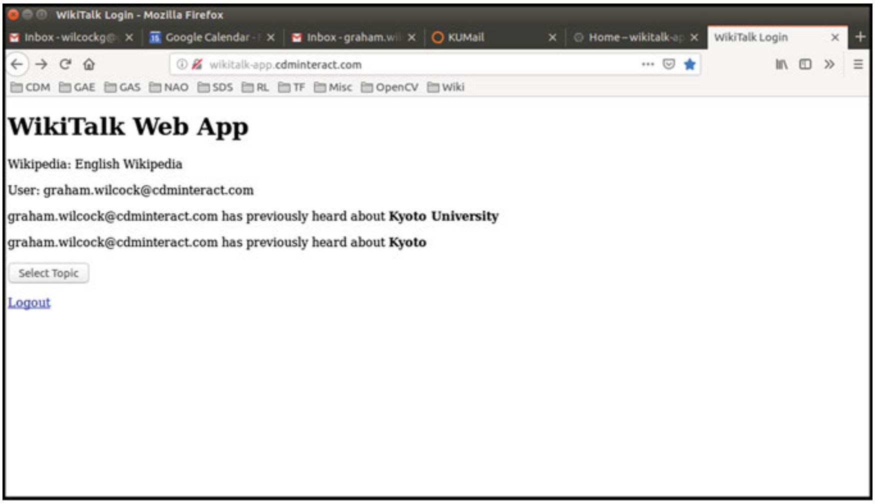


Fig. 2 WikiTalk web app showing a user's recent interaction history

3 Robots that Listen

Recently an *attentive listening* role has received attention as a task for social robots. For example, the ERICA android robot [8] adopts this role to encourage people, especially senior people, not only to talk but also to keep talking over longer periods in order to maintain communication ability and mental health. To allow users to keep talking smoothly, work on this task focuses on backchanneling and filler generation, especially on timing, prosody and synchrony [9]. To encourage users to make more informative and longer contributions, simple questions are generated by repeating words said by the the user [12] without using any source of world knowledge.

The attentive listening role can enhance the social robot's capability to take part in natural interaction. For WikiTalk, we propose this capability through *Wikipedia-based listening* based on wikification of speech. Linking entities to related Wikipedia articles [13, 15] can already be done effectively for written texts, but so far it has been difficult for spoken dialogues.

Recently however, wikification of speech has become more feasible. For example, Kim et al. [11] propose wikification of concept mentions in spoken dialogues using domain constraints from Wikipedia. They identify key differences between texts and spoken dialogues, and propose wikification of spoken dialogue using classifiers to analyze dialogue-specific aspects of a given mention, and ranking filtered candidates to identify the concept most relevant to the mention. Milde et al. [14] also demonstrate wikification of speech in their Ambient Search engine.[2] A demo video[3] is impressive, but the demo uses only the smaller Simple English Wikipedia. When we experimented with the system the ranking of linked articles was puzzling, due perhaps to the limited content and fewer connections between the articles.

Human-robot interaction will be greatly enhanced if robots are able to follow the constantly changing topics in human conversation, even to a limited extent, by means of Wikipedia-based listening. We can call this a *WikiListen* capability. Then WikiTalk and other Wikipedia-based talking systems will enable robots to join in conversations with topically relevant contributions.

By combining Wikipedia-based listening with long-term relations based on face recognition and storage of user interaction histories, robots will not only be able to join in conversations in the short term, but will also remember what topics different people talked about earlier, so they will be able to resume those topics when they meet and interact with those people on future occasions.

Acknowledgements The first author thanks Prof. Tatsuya Kawahara of Kyoto University for the opportunity to participate in the ERICA robot project. The second author acknowledges the support of the New Energy and Industrial Technology Development Organisation (NEDO) in Japan.

[2] https://github.com/bmilde/ambientsearch.

[3] https://raw.githubusercontent.com/bmilde/ambientsearch/master/demo_video_august_2016.mp4.

References

1. Amos B, Ludwiczuk B, Satyanarayanan M (2016) OpenFace: a general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU
2. Fu Z (2018) A paper list for style transfer in text. [GitHub]. https://github.com/fuzhenxin/Style-Transfer-in-Text
3. Jokinen K (2015) Bridging gaps between planning and open-domain spoken dialogues. In: Gala N, Rapp R, Bel-Enguix G (eds) Language Production, Cognition, and the Lexicon. Springer, Berlin, pp 347–360
4. Jokinen K, Fukuda K, Iino N, Nishimura S, Nishimura T, Oota Y, Watanabe K, Yoshida Y (2019) Ethical and privacy issues in interactive service applications for elder people concerning dialogues and social robots. In: 7th Serviceology Conference. Tokyo
5. Jokinen K, Nishimura S, Watanabe K, Nishimura T (2018) Human-robot dialogues for explaining activities. In: 9th International Workshop on Spoken Dialogue Systems (IWSDS 2018). Singapore
6. Jokinen K, Wilcock G (2012) Multimodal signals and holistic interaction structuring. In: Proceedings of 24th International Conference on Computational Linguistics (COLING 2012). Mumbai, pp 527–538
7. Jokinen K, Wilcock G (2014) Multimodal open-domain conversations with the Nao robot. In: Mariani J, Rosset S, Garnier-Rizet M, Devillers L (eds) Natural Interaction with Robots, Knowbots and Smartphones: Putting Spoken Dialogue Systems into Practice. Springer, Berlin, pp 213–224
8. Kawahara T (2018) Spoken dialogue system for a human-like conversational robot ERICA. In: 9th International Workshop on Spoken Dialogue Systems (IWSDS 2018). Singapore
9. Kawahara T, Uesato M, Yoshino K, Takanashi K (2015) Toward adaptive generation of backchannels for attentive listening agents. In: Proceedings of the 6th International Workshop on Spoken Dialogue Systems (IWSDS 2015). Busan, South Korea
10. Kazemi V, Sullivan J (2014) One millisecond face alignment with an ensemble of regression trees. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1867–1874
11. Kim S, Banchs RE, Li H (2015) Wikification of concept mentions within spoken dialogues using domain constraints from wikipedia. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pp 2225–2229. http://www.aclweb.org/anthology/D15-1199
12. Lala D, Milhorat P, Inoue K, Ishida M, Takanashi K, Kawahara T (2017) Attentive listening system with backchanneling, response generation and flexible turn-taking. In: Proceedings of the SIGDIAL 2017 Conference. Saarbrücken, Germany, pp 127–136
13. Mihalcea, R., Csomai, A.: Wikify! Linking documents to encyclopedic knowledge. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM 2007), pp. 233–242. Lisbon (2007)
14. Milde B, Wacker J, Radomski S, Muhlhäuser M, Biemann C (2016) Ambient search: a document retrieval system for speech streams. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations. Osaka
15. Milne D, Witten I (2008) Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008). Napa Valley, pp 509–518
16. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 815–823
17. Wilcock G, Jokinen K (215) Multilingual WikiTalk: Wikipedia-based talking robots that switch languages. In: Proceedings of the 16th Annual SIGdial Meeting on Discourse and Dialogue. Prague

A Classification-Based Approach to Automating Human-Robot Dialogue

Felix Gervits, Anton Leuski, Claire Bonial, Carla Gordon, and David Traum

Abstract We present a dialogue system based on statistical classification which was used to automate human-robot dialogue in a collaborative navigation domain. The classifier was trained on a small corpus of multi-floor Wizard-of-Oz dialogue including two wizards: one standing in for dialogue capabilities and another for navigation. Below, we describe the implementation details of the classifier and show how it was used to automate the dialogue wizard. We evaluate our system on several sets of source data from the corpus and find that response accuracy is generally high, even with very limited training data. Another contribution of this work is the novel demonstration of a dialogue manager that uses the classifier to engage in multi-floor dialogue with two different human roles. Overall, this approach is useful for enabling spoken dialogue systems to produce robust and accurate responses to natural language input, and for robots that need to interact with humans in a team setting.

F. Gervits (✉)
Tufts University, Medford, MA, USA
e-mail: felix.gervits@tufts.edu

A. Leuski · C. Gordon · D. Traum
USC Institute for Creative Technologies, Playa Vista, CA, USA
e-mail: leuski@ict.usc.edu

C. Gordon
e-mail: cgordon@ict.usc.edu

D. Traum
e-mail: traum@ict.usc.edu

C. Bonial
U.S. Army Research Laboratory, Adelphi, MD, USA
e-mail: claire.n.bonial.civ@mail.mil

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_10

1 Introduction

A major drive in human-robot interaction (HRI) research is to enable robots to serve as genuine partners in teams with humans. Such heterogeneous teams are intended for use in a variety of applications including classroom tutoring, disaster-relief, and military reconnaissance. In particular, there has been a great deal of research in task-oriented remote communication for the purpose of urban search and rescue (USAR). HRI is desirable for these domains, as a robot can be used to explore a hazardous area while a human monitors the situation and gives instructions remotely.

A critical requirement of effective teaming in collaborative USAR domains is communication [7, 19]. Humans use communication to share task-relevant information, give instructions, discuss plans, and many other functions. As a result, robots will need to handle at least some of these functions if they are expected to fill the role of a human teammate. At minimum, robots will need to interpret a command in the form of speech input, perform the corresponding action of the command, and produce a feedback response to the human. This involves bidirectional communication in which the robot not only takes orders but also responds in meaningful ways. Error handling and dialogue management are additional requirements needed for more robust interactions. Finally, naturalness and flexibility of the system are also desirable: it is important that humans can talk to the robot in a natural manner, which includes all the disfluencies and irregularities that arise in natural language (NL), and the robot should be robust to variability in speech in order to serve as a more effective conversational partner.

1.1 Background and Related Work

In order to meet the above requirements, various types of dialogue models have been proposed and attempted. The most basic is finite-state systems in which dialogues are represented as a pre-determined state transition network [18]. Finite-state systems are effective for small, highly-structured domains in which the flow of dialogue is known in advance. However, such systems are generally inflexible to input that is not in the network, and do not seem well suited to the complex USAR domains of interest. Frame-based dialogue models have also been proposed, which involve filling in various slots in a "form" corresponding to an action or utterance [26]. These offer more flexibility to handle increasingly complex dialogues, but struggle with utterances that do not fit into a frame. As a result, frame-based models have been mainly used in tasks with a fixed set of slots, such as travel booking [8]. Finally, plan-based systems turn dialogue into a planning problem in which a human's utterance is mapped to a speech act and the system performs logical inference over its beliefs and goals in order to select an appropriate response [1, 4]. While this kind of model is very useful for handling complex dialogues, it relies on the difficult problem of identifying speech acts and intentions.

We adopt the *corpus-based robotics* approach, wherein the system is trained on corpus data from the target domain [3]. The corpus used for training could involve human-human instruction [6], human-robot instruction [17], or human-robot instruction in a Wizard-of-Oz (WoZ) paradigm [2, 5]. The corpus data includes examples of natural commands that robots will need to interpret and act on, and serves as a source of interaction patterns to inform dialogue management policies. Through the use of statistical techniques, systems using this approach have been very effective at modeling various aspects of dialogue [20, 24, 25]. This approach also offers flexibility, as data-driven models are often robust to noisy and disfluent data. However, a major drawback with many machine learning techniques is that large, annotated data sets are required for training [21]. Such data sets are often unavailable or infeasible to produce due to the large cost and effort needed to collect, transcribe, and annotate data. Moreover, new ones are often needed for each target domain because the systems do not usually extend beyond the particular training domain.

1.2 Motivation and Present Work

We are currently developing an end-to-end spoken dialogue system for use in a collaborative human-robot navigation domain. The system is trained on a small corpus involving a dual WoZ setup in which one wizard handles the dialogue management (DM) and the other handles robot navigation (RN). The rationale behind using such a corpus is that we wanted the system to interpret speech and respond in an appropriate, human-like manner. This approach provides data-driven insights into what such a response would be, and what variety we should expect, in the context of a collaborative navigation task. Our ultimate goal is to create a fully autonomous robot. In this paper we describe initial attempts to automate the natural language dialogue capabilities using a statistical classifier based on cross-language information retrieval. The system operates across multiple floors (i.e., distinct communication channels) and "translates" messages between the human user and the RN component or wizard, and gives positive and negative feedback to the human user.

Given our small corpus, we were interested in exploring how far we can get with a data-driven approach using such limited training data and limited annotation. Most end-to-end systems require large training sets to get reasonable performance, but previous evaluations of a similar classifier have shown reasonably high accuracy with only a few hundred utterances for training [9] compared to the hundreds of thousands needed in other systems (e.g., [20]). Note that we do not claim that our approach is immune to the limitations of other data-driven systems, and we discuss some of these limitations in Sect. 5. However, the goal is to mitigate some of these limitations through our classification and DM approach.

Below, we introduce our task domain and provide details of the corpus used. Next, we describe our classification approach as well as the DM policies that were implemented. Finally, we evaluate our system on several data sets of varying size from the corpus to compare response accuracy. In the evaluation, the following points will be addressed: (1) accuracy of the classifier (especially as it relates to the size and composition of the training data), (2) adequacy of the DM response, and (3) integration of the system in a robotic architecture.

2 Collaborative Human-Robot Navigation Task

2.1 Task Domain

Our task domain involves collaborative navigation akin to a USAR scenario. In the task, a human serves as a *Commander* and supervises a remotely-located robot to perform a navigation task in an unfamiliar physical environment. The environment is modeled after a house and includes various rooms and objects consistent with this environment type (rooms, hallways, etc.). The goal of the task is to work together as a team to accomplish two subtasks—one related to searching (e.g., locate shoes) and one related to analysis (e.g., evaluate whether the area can serve as a headquarters).

Throughout the task, the Commander is seated in front of a computer with an interface showing task-relevant information. The interface includes a 2D occupancy grid showing the robot's location, a snapshot of the last image taken by the robot, and a textbox showing the robot's dialogue responses (see top-right of Fig. 1). To direct the robot, the Commander is able to speak freely using unconstrained natural language. Examples of common instructions include "Move forward 10 feet", "Take a picture", and "Turn right 45°". People also used landmark-based instructions such as "Move to face the yellow cone", and "Go to the doorway to your right", although these were less common than the metric-based instructions [15].

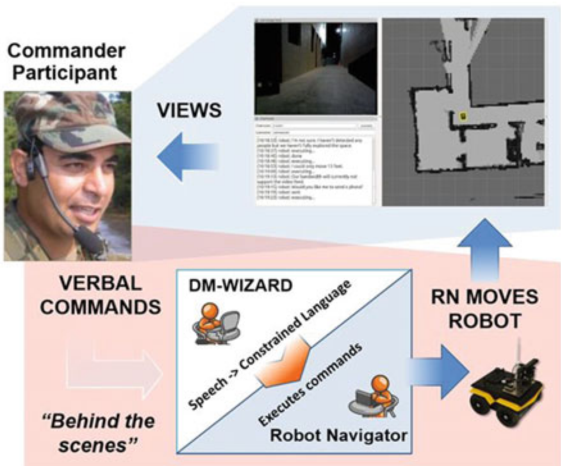


Fig. 1 Experimental task domain with dual-wizard setup (from [13])

The task was run using a dual-WoZ setup wherein one wizard controlled the DM and the other controlled the RN. Importantly, the wizards had to communicate with one another to ensure that actions and responses were performed correctly and in a timely manner [14]. The task was run over several experiments, with additional experiments currently in progress. In Experiment (Exp.) 1, the DM-Wizard typed free responses to the Commander and RN-Wizard according to pre-established guidelines (see [13]). From this, we developed a GUI that was used by the DM-Wizard in Exp. 2 to provide quicker and more uniform responses [2, 16]. The same GUI was used in Exp. 3, except that here we used a simulated robot and environment rather than a physical one. Exp. 1 and 2 had 10 participants each, whereas Exp. 3 had 62 participants.

2.2 *Corpus and Annotation*

A corpus was created from the Exp. 1 and 2 data (annotation for Exp. 3 is still in progress). Dialogues were annotated according to the scheme described in [23], which was specifically designed to handle the multiple conversational floors in our dual-wizard setup. These floors include: (1) Commander and DM-Wizard and (2) DM and RN-Wizards. The main unit of dialogue in our annotation scheme is the *transaction unit (TU)*, which includes the initial utterance expressing the intent of the speaker and all subsequent utterances across all floors that are used to achieve the intent of the original speaker. An example TU can be found in Table 1. In addition, our scheme also includes three distinct types of *relations*, which are used to characterize

Table 1 Example TU and annotation from the corpus. The * indicates that the antecedent is part of a sequence of expansions

	Left floor		Right floor		Annotation	
#	Commander	DM ->Commander	DM ->RN	RN	Ant.	Rel.
1	Rotate to the right ninety degrees					
2	And take a photo				1	Continue
3		Ok			2*	Ack-understand
4			Turn right 90°		1	Translation-r
5			Then...		4	Link-next
6			Send image		2	Translation-r
7				Done and sent	6*	Ack-done
8		Done, sent			7	Translation-l

how an utterance is related to an antecedent (previous utterance). These relation types include *expansions*, *responses*, and *translations* along with various subtypes of each. Expansions are continuations of a previous utterance by the same speaker in the same floor. Responses are produced by different speakers in the same floor, and include several types of acknowledgements, clarifications, and answers. Finally, translations are used to relate utterances in different floors, and include two subtypes: *translation-right* involves the DM-Wizard translating a Commander's instruction to the RN-Wizard for action (e.g., "Move forward three feet"), and *translation-left* involves the DM-Wizard translating the RN-Wizard's action to the Commander in the form of feedback (e.g., "I moved forward three feet"). In total, the corpus included 2230 TUs across 60 dialogues from 20 different Commanders (each Commander participated in three dialogues) [23].

3 Natural Language Dialogue Processing

In this section, we provide an overview of the NL approach toward mimicking the DM-Wizard's utterance selection policies based on input from Commander instructions. We first outline the classification approach and describe the data processing that we carried out on the corpus data. We then describe the DM policies that were implemented to make use of the classifier output in order to produce appropriate responses across the multiple floors. Finally, we evaluate the output on new test data from Exp. 3.

3.1 Classifier Approach

The task of the language classifier is to analyze the Commander's instruction and select the appropriate utterances from the system's collection of responses. It involves the following three step process:

First, the classifier indexes the existing language data—a dataset of instruction-response pairs that we have collected during WoZ experiments. It generates a statistical language model for each natural language utterance (both instruction and response): $P(w|W)$, where w is a word in the vocabulary and W is the utterance. Note that the vocabularies of instructions and responses are different.

Next, the classifier uses the indexed data to construct a joint model of the instructions and responses, and to compute the cross-language relevance model for each new Commander's instruction $P(w|C)$, where w is a word in the response vocabulary and C is the instruction. Please see our previous paper [11] for the technical details of the approach.

Finally, the classifier compares the language model $P(w|C)$ with the language model of each response in the system's dataset, $P(w|R_i)$. Here R_i is the ith response in the dataset. It returns all the responses with the similarity score above a predefined threshold. The threshold is determined during the classifier training phase.

The classifier implementation is part of the NPCEditor platform, which has been used in the past to build effective question-answering conversational systems [10]. The approach requires a relatively small amount of training data, has a small number parameters to tune (three parameters, including the threshold, in most cases), and is robust to noise and errors in the input [9]. Next, we explain how we processed our experimental data to train the classifier.

3.2 Data Processing

Instruction-Response pairs The first step was to constrain the multi-floor data to something closer to what the classifier uses in terms of linked initiative-response pairs. This is challenging because in our data there are two different types of reactions to a Commander input: *responses* to the Commander (including positive and negative feedback) and *translations* of actionable Commander instructions to the RN. To create this dataset, we first used a script to parse the annotated corpus data and link each utterance produced by the Commander with the DM-Wizard's responses to it. We did this for several relation types, including translation-right, and several response subtypes (clarification, acknowledgment, answer, etc.); this resulted in a set of instruction-response pairs. For example, "Take a picture" → "image" is an example of a *translation-right* pair, in which a Commander's instruction is translated into a shorthand request sent to the RN-Wizard, and "Move forward" → "Please tell me how far to move forward" is an example of a *request-clarification* pair, in which the Commander's open-ended instruction prompts a clarification request from the DM-Wizard.

Coherence rating Since the resulting set of instruction-response pairs were automatically generated from scripts, the next step involved filtering these pairs for coherence. We used the 4-point coherence rating scale from [22], where a 1 represents a response that is either missing or irrelevant to the instruction, a 2 represents a response that relies on external context to match, a 3 represents a response that indirectly addresses the instruction but that contains additional (irrelevant) information, and a 4 represents a response that directly addresses the instruction. Using this rating scale, we manually inspected the instruction-response pairs from Exp. 1 and 2 and rated each one. In Exp. 1, out of a total of 999 pairs, 96 pairs had a rating of 1, 222 pairs had a rating of 2, 1 pair had a rating of 3, and 680 pairs had a rating of 4. In Exp. 2, out of a total of 1419 pairs, 50 pairs had a rating of 1, 387 pairs had a rating of 2, 4 pairs had a rating of 3, and the remaining 978 pairs had a rating of 4. For the final training set, we included all the pairs that had a 3 or 4 coherence rating.

Data smoothing Finally, we conducted a smoothing step in order to ensure that the training data would cover the various fields in the most common instructions. For example, a command such as "Move forward four feet" might not exist in the corpus, but is nonetheless an instruction that the system should be able to carry out. In order to capture these missing fields, we added a set of 250 pairs to the training data. 196 of these “smoothing pairs” came from the table which was used to generate the Exp. 2 GUI (see [2]). This ensured that the system could at least interpret the most common actionable commands from the experimental studies. The remaining 54 pairs were hand-generated to fill in values that were missing from the corpus data, and simply included additional values for the existing commands.

3.3 Dialogue Manager Policies

After being trained on the instruction-response pairs as described above, the classifier learned a mapping between commands and responses from the data. We then implemented a DM whose role it was to use the classifier output to select appropriate responses and send them to the corresponding floor (Commander or RN).

The DM works in the following manner. First, it receives an utterance in the form of a string after it has passed through the speech recognizer. The classifier then ranks the top responses that match the instruction and sends this list back to the DM. Upon receiving the matching response from the classifier, the DM then sends this to the corresponding floor. Actionable commands are formatted and sent to the RN whereas the corresponding feedback message (“Executing”, “Moving”, “Turning”, etc.) is sent to the Commander’s interface. Non-actionable commands cause the system to generate a response (usually a type of clarification or acknowledgment) to the Commander in order to repair the instruction.

Some specific policies were implemented to handle problematic input. One such policy handles the case when the classifier finds no match. This usually means that the command was outside of the domain, or that any potential matches were below threshold. In either case, the DM will cycle through several general clarification requests when this happens, prompting the Commander to repeat and reformulate the instruction. Another policy was implemented to handle cases in which the classifier selected multiple responses. In this case, it always picks the one with the highest score, but in the case of a tie, a random response is chosen from the tied options.

4 Evaluation

The DM in combination with the trained classifier allowed us to replicate many of the dialogue behaviors from the experimental data. To evaluate the performance of our system, we trained six classifiers using varying amounts of source data from each of the first two experiments (Exp. 1 and Exp. 2). Training data for each classifier consisted of annotated user utterances from a given experiment, which were processed using the methods described in Sect. 3.2. Additional smoothing data was added to each of the three original classifiers in order to test for the benefit of including these extra pairs. A summary of the combinations of data, as well as the number of training utterances, responses and links between the two is presented in Table 2. The "# links" column refers to the number of connections between utterances and responses in that set. These connections do not represent a one-to-one relationship, as any given utterance can be linked to more than one response, and vice versa. We also evaluated the DM separately from the classifier to test the appropriateness of responses produced by the system.

4.1 Classifier Evaluation and Results

Each of the six trained classifiers was tested on a test set comprised of three previously unseen dialogues that were randomly selected from Exp. 3. These dialogues were annotated, and the instruction-response pairs were extracted, but no other processing was done on these pairs as we sought to maintain the raw data for testing. In total, the test set included 183 instruction-response pairs.

For each utterance in the test set, we compared the best classifier match to the expected output, which is the one actually produced by the DM-Wizard in the test data. Accuracy was calculated as the percentage of queries where the best classifier match is the expected response. The results of our evaluation are displayed in the right-most column of Table 2. Accuracy scores ranged from 61% in the Exp. 2 classifier to 75% in the combined Exp. 1+2 classifier with added smoothing pairs. In general, we found that performance improved with the addition of the smoothing pairs, and this improvement ranged from 2.7 to 5.5% depending on the original training set. As expected, this largely benefited the Exp. 1 and Exp. 2 classifiers, which had limited data and were missing many of the basic pairs that were part of the smoothing set. Interestingly, we found that accuracy on the unconstrained Exp. 1 data (65%) was higher than the GUI-based Exp. 2 data (61%). This is likely due to the reduced number of responses in Exp. 2 caused by the standardization of the GUI. Without the smoothing data added, there may not have been enough unique responses to match the test queries. Overall, the highest accuracy (75%) was obtained for the classifier trained on all the data. This is a promising result, and suggests that relatively high accuracy can be achieved with under 1000 utterances of training.

4.2 Dialogue Adequacy Evaluation

It is important to note that classifier accuracy is only part of what we are interested in. Perfect matches are of course desirable, however a response can still be reasonably appropriate even if not an exact match of the corpus data (e.g., "turn 20°" vs "turn 25°"). In order to evaluate the classifier in terms of expected impact on the dialogue, we examined the 45 (about 25% of test set) utterances that the combined classifier got wrong in the previous evaluation. For each of these responses, we placed them into

Table 2 Classifier data summary. Accuracy represents the proportion of the classifier's responses that matched the test query

Training data	Test data	#Utterances	#Responses	#Links	Accuracy
Exp. 1 only	Exp 3	347	247	366	0.6503
Exp. 1 + smoothing	Exp 3	593	436	614	0.6831
Exp. 2 only	Exp 3	424	141	429	0.6066
Exp. 2 + smoothing	Exp 3	670	328	675	0.6612
Exp. 1 & 2	Exp 3	722	303	751	0.7268
Exp. 1 & 2 + smoothing	Exp 3	966	483	995	**0.7541**

Table 3 Dialogue adequacy evaluation showing the type and relative frequency of the 45 system responses that did not match the test set

	Felicitous	Approximate	Context-dependent	Wrong	No response
Instruction	Turn one eighty	Go west five feet	Go to plant	Go back to table	Rotate toward camera towards calendar
Test-set response	Turn 180	Turn to face west; move forward 5 feet	Go to Dark room plant	Move back towards table	Move to conference calendar
DM response	Rotate 180	Turn to face west; move forward 10 feet	Go to Foyer plant	Return to starting point	< no response >
Count (out of 45)	8	15	14	7	1
Proportion	0.18	0.33	0.31	0.16	0.02

one of five categories: *Felicitous*—appropriate responses that would have the same effect as the correct response, *Approximate*—responses that differed only slightly from the correct one (e.g., variation in turn radius or movement distance), *Context-Dependent*—responses that could be correct, but that depend on the context in which they occurred, *Wrong*—responses that were not appropriate for the given command, and *No Response*—indicating that the classifier did not find a match. Table 3 summarizes the analysis of responses that did not match the test-set, including examples of each type and the frequency of each.

Felicitous responses are expected to have no negative impact on the dialogue. Approximate responses might have a small delay to extend or correct the robot's behavior. Wrong responses are expected to have a more severe impact in terms of either cancelling the instruction mid-operation, or undoing it after. Context-Dependent responses might either have no negative effect (like Felicitous responses) or a negative effect (like Wrong responses), depending on how the context is applied to create a full instruction. When the classifier does not find a match, the DM instructs the user to repeat or rephrase the previous instruction, slowing down the dialogue, but not impacting the robot's physical situation.

In our analysis, we found that over half of the incorrect responses in the test set were either *Felicitous* or *Approximate* to the correct response. This suggests that, despite not matching the test data, these responses would still be appropriate and would advance the dialogue. Only one case had no response, and the remaining cases were split between the *Context-Dependent* and *Wrong* categories. Fortunately, these responses were infrequent, representing only 11% of the total test set.

4.3 Demonstration: Integration in the ScoutBot Architecture

One of the primary goals of this research project is to develop a fully automated end-to-end dialogue system for collaborative navigation. To that end, we (and colleagues) have implemented a system called ScoutBot, which was designed to automate the tasks of the dual wizards in our navigation task [12]. We have found in pilot testing that Scoutbot can effectively interpret simple instructions and navigate accordingly, but a more detailed evaluation is work in progress. Currently, the main limitation of Scoutbot is the inability to handle landmark-based instructions such as "Move from A to B". Addressing this will require additional mechanisms (see below), but importantly, the system still works well for the majority of examples and should be sufficient for the team to complete the task. A demonstration video of ScoutBot can be found at the following link: http://people.ict.usc.edu/~traum/Movies/scoutbot-acl2018demo.wmv.

5 Conclusion and Future Work

The ability to converse with robots is an important part of human-robot teaming envisioned for many applications. As a result, end-to-end dialogue systems that facilitate effective communication are becoming increasingly needed. We presented a data-driven system to achieve this goal that uses a statistical classifier and dialogue manager to interpret natural language commands, produce appropriate responses, and carry out actions. In our evaluation, the system was shown to maintain relatively high response accuracy even with limited and noisy training data.

Moving forward, we are in the process of extending the system to handle some of the limitations we encountered, namely landmark-based instructions and complex, multi-turn commands. The former will require a context model in which the system tracks the robot's location throughout the map and biases the DM to favor objects and locations in the local context. A possible solution for the latter is supplementing our system with an information extraction approach in which the key parameters of a command (e.g., action, distance, etc.) are extracted and used to fill a semantic frame. This will also enable us to provide more detailed clarification requests to obtain specific pieces of information (e.g., "how far should I move forward?"). Finally, we expect the additional data from Exp. 3 to further improve the classifier accuracy and reduce the number of incorrect responses. Overall, this approach offers a practical alternative to those that require large-scale corpora for dialogue systems, and shows that good performance is possible with a data-driven approach using a smaller data set.

Acknowledgements This research was sponsored by the U.S. Army Research Laboratory and by a NASA Space Technology Research Fellowship for the first author.

References

1. Allen JF, Perrault CR (1980) Analyzing intention in utterances. Artif Intell 15(3):143–178
2. Bonial C, Marge M, Foots A, Gervits F, Hayes CJ, Henry C, Hill SG, Leuski A, Lukin SM, Moolchandani P, Pollard KA, Traum D, Voss CR (2017) Laying down the yellow brick road: development of a wizard-of-oz interface for collecting human-robot dialogue. In: Symposium on natural communication for human-robot collaboration, AAAI FSS
3. Bugmann G, Klein E, Lauria S, Kyriacou T (2004) Corpus-based robotics: a route instruction example. In: Proceedings of intelligent autonomous systems, pp 96–103
4. Cohen PR, Perrault CR (1979) Elements of a plan-based theory of speech acts. Cogn Sci 3(3):177–212
5. Dahlbäck N, Jönsson A, Ahrenberg L (1993) Wizard of oz studies - why and how. Knowl-Based Syst 6(4):258–266
6. Eberhard KM, Nicholson H, Kübler S, Gundersen S, Scheutz M (2010) The indiana "cooperative remote search task" (crest) corpus. In: Proceedings of LREC 2010
7. Gervits F, Eberhard K, Scheutz M (2016) Team communication as a collaborative process. Front Robot AI 3:62
8. Issar S, Ward W (1993) CMLPs robust spoken language understanding system. In: Third European conference on speech communication and technology
9. Leuski A, Traum D (2008) A statistical approach for text processing in virtual humans. In: Proceedings of the 26th army science conference
10. Leuski A, Traum D (2011) NPC editor: creating virtual human dialogue using information retrieval techniques. AI Mag 32(2):42–56
11. Leuski A, Traum DR (2010) Practical language processing for virtual humans. In: IAAI-10
12. Lukin SM, Gervits F, Hayes CJ, Moolchandani P, Leuski A, Rogers III JG, Amaro CS, Marge M, Voss CR, Traum D (2018) ScoutBot: a dialogue system for collaborative navigation. In: Proceedings of ACL
13. Marge M, Bonial C, Byrne B, Cassidy T, Evans AW, Hill SG, Voss C (2016) Applying the wizard-of-oz technique to multimodal human-robot dialogue. In: Proceedings of RO-MAN
14. Marge M, Bonial C, Pollard KA, Artstein R, Byrne B, Hill SG, Voss C, Traum D (2016) Assessing agreement in human-robot dialogue strategies: a tale of two wizards. In: International conference on intelligent virtual agents. Springer, pp 484–488
15. Marge M, Bonial C, Foots A, Hayes C, Henry C, Pollard K, Artstein R, Voss C, Traum D (2017) Exploring variation of natural human commands to a robot in a collaborative navigation task. In: Proceedings of the first workshop on language grounding for robotics, pp 58–66

16. Marge M, Bonial C, Lukin S, Hayes C, Foots A, Artstein R, Henry C, Pollard K, Gordon C, Gervits F, Leuski A, Hill S, Voss C, Traum D (2018) Balancing efficiency and coverage in human-robot dialogue collection. In: Proceedings of AI-HRI AAAI-FSS
17. Marge MR, Rudnicky AI (2011) The teamtalk corpus: route instructions in open spaces. In: Proceedings of SIGdial
18. McTear MF (1998) Modelling spoken dialogues with state transition diagrams: experiences with the CSLU toolkit. In: Fifth international conference on spoken language processing
19. Murphy R (2004) Human-robot interaction in rescue robotics. IEEE Trans Syst Man Cybern Part C 34(2):138–153
20. Serban IV, Sordoni A, Bengio Y, Courville AC, Pineau J (2016) Building end-to-end dialogue systems using generative hierarchical neural network models. In: AAAI, vol 16, pp 3776–3784
21. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2018) A survey of available corpora for building data-driven dialogue systems: the journal version. Dialogue Discourse 9(1):1–49
22. Traum D, Georgila K, Artstein R, Leuski A (2015) Evaluating spoken dialogue processing for time-offset interaction. In: Proceedings of SIGdial, pp 199–208
23. Traum DR, Henry C, Lukin SM, Artstein R, Pollard KA, Bonial C, Lei S, Voss CR, Marge M, Hayes C, Hill S (2018) Dialogue structure annotation for multi-floor interaction. In: Proceedings of LREC
24. Vinyals O, Le Q (2015) A neural conversational model. arXiv:1506.05869
25. Wen TH, Gasic M, Mrksic N, Su PH, Vandyke D, Young S (2015) Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In: Proceedings of EMNLP
26. Xu W, Rudnicky AI (2000) Task-based dialog management using an agenda. In: Proceedings of the 2000 ANLP/NAACL workshop on conversational systems, vol 3, pp 42–47

Engagement-Based Adaptive Behaviors for Laboratory Guide in Human-Robot Dialogue

Koji Inoue, Divesh Lala, Kenta Yamamoto, Katsuya Takanashi, and Tatsuya Kawahara

Abstract We address an application of engagement recognition in human-robot dialogue. Engagement is defined as how much a user is interested in the current dialogue, and keeping users engaged is important for spoken dialogue systems. In this study, we apply a real-time engagement recognition model to laboratory guide by autonomous android ERICA which plays the role of the guide. According to an engagement score of a user, ERICA generates adaptive behaviors consisting of feedback utterances and additional explanations. A subject experiment showed that the adaptive behaviors increased both the engagement score and related subjective scores such as interest and empathy.

1 Introduction

Spoken dialogue systems are expected to realize social interaction with real users in more varied scenarios. Conventional systems were applied to scenarios such as museum guide [21] and mental diagnosis [3]. We have developed a spoken dialogue system for the autonomous android ERICA [11, 12]. Giving specific social roles to ERICA, we aim to realize natural dialogue between ERICA and users. We have

K. Inoue (✉) · D. Lala · K. Yamamoto · K. Takanashi · T. Kawahara
Graduate School of Informatics, Kyoto University, Kyoto, Japan
e-mail: inoue@sap.ist.i.kyoto-u.ac.jp

D. Lala
e-mail: lala@sap.ist.i.kyoto-u.ac.jp

K. Yamamoto
e-mail: yamamoto@sap.ist.i.kyoto-u.ac.jp

K. Takanashi
e-mail: takanashi@sap.ist.i.kyoto-u.ac.jp

T. Kawahara
e-mail: kawahara@sap.ist.i.kyoto-u.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_11

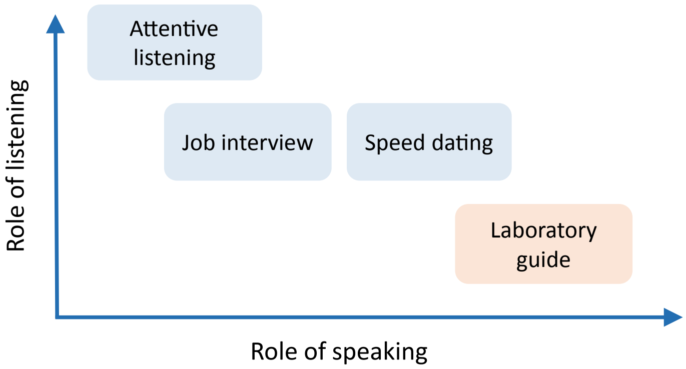


Fig. 1 Social roles expected to ERICA

considered several social roles so far by taking into account two factors of ERICA in dialogue: speaking and listening as depicted in Fig. 1. Focusing on the role of listening, we implemented an attentive listening system [14] and a job interview dialogue system [7] for ERICA. In this study, we focus on the role of speaking and implement a spoken dialogue system for laboratory guide where ERICA explains about a laboratory to users. In this scenario, the majority of dialogue is explanations from guides. However, it is needed to not only just explain but also recognize the listening attitude of visitors. A human-like laboratory guide is expected to dynamically change the explanation and its behaviors according to states of users so that it increases the quality of user experience in the dialogue.

We address engagement which represents the process by which dialogue participants establish, maintain, and end their perceived connection to one another [19]. This concept is practically defined as how much a user is interested in the current dialogue [24]. Therefore, it is important for spoken dialogue systems to make users engaged in dialogue. Engagement recognition has been widely studied using mainly non-verbal behaviors [1, 2, 15, 17, 18, 23, 26]. We also studied engagement recognition by utilizing listener behaviors such as backchannels, laughing, head nodding, and eye gaze [9, 10]. Furthermore, we implemented real-time engagement recognition by detecting the above listener behaviors automatically [8]. On the other hand, fewer studies have been made on how to manage system behaviors after the system recognizes user engagement [20, 23, 25, 26].

In this study, we utilize the real-time engagement recognition model in the laboratory guide by ERICA. According to the engagement level of a user, ERICA generates adaptive behaviors to keep or increase the engagement level itself. Furthermore, we investigate the subjective evaluation of engagement together with rapport that would be affected by the engagement-based adaptive behaviors. This study aims to confirm the effectiveness of engagement recognition in a social scenario of human-robot dialogue.

This paper is organized as follows. The real-time engagement recognition model is introduced in Sect. 2. The adaptive behaviors in the context of a laboratory guide are explained in Sect. 3. A user experiment is conducted in Sect. 4. Section 5 concludes this paper with future direction.

2 Engagement Recognition Based on Listener Behaviors

We addressed engagement recognition based on listener behaviors such as backchannels, laughing, head nodding, and eye gaze. The listener behaviors are non-linguistic cues so that engagement recognition can be independent of dialogue content, which makes it robust and applicable to any scenario. The engagement recognition was done during each system's dialogue turn when a user was being a listener. Each listener behavior was defined as an occurrence of the behavior, and the input feature was represented as the combination of the binary values, termed a *behavior pattern*. Note that the eye-gaze behavior was defined as an event if the user looked at the robot's face longer than a certain time (10 s). Finally, the engagement recognition model outputs an engagement score for each system's turn so we would be able to utilize the score to decide a system action for the next system's turn.

In previous works, we proposed a latent character model for engagement recognition [8, 10]. Since the perception of engagement is subjective, oracle labels of engagement depend on perceivers (annotators). Our approach is based on a hierarchical Bayes model and introduces latent variables, called character, to represent the difference among annotators. Figure 2 illustrates the graphical model of the proposed model that contains two kinds of parameters to be estimated in the training phase: a character distribution of each annotator and an engagement distribution. In the test phase, we calculate the probability of the engaged label of a target annotator by

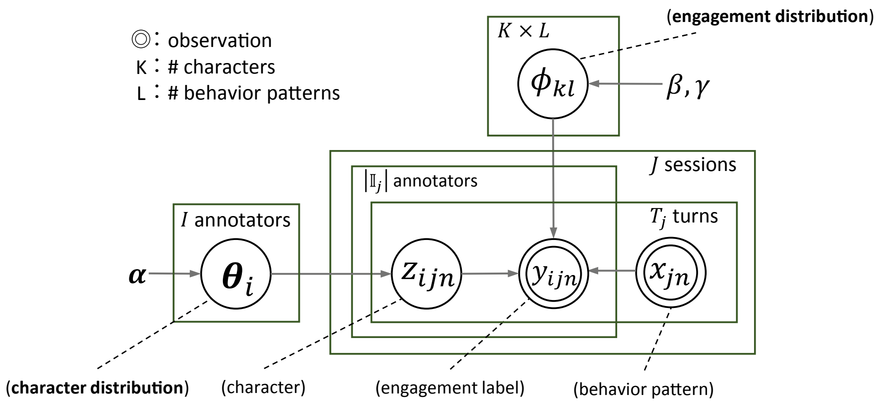


Fig. 2 Graphical model of latent character model

using both the engagement distribution and the target annotator's character distribution. The advantage is that our model can simulate each annotator's perception by using the corresponding character distribution. Therefore, our model can recognize each annotator's label more precisely. We conducted a manual annotation where each annotator gave a binary label (engaged or not) in each dialogue turn. Experimental results showed that our model achieved an accuracy of 71.1% which was higher than those of other methods that did not take into account the character variables. Furthermore, we examined the relationship between the estimated character distribution and a personality trait (Big Five) of each annotator. We calculated regression coefficients from Big Five scores to parameters of the character distribution. Using this regression result, if we specify a personality trait score expected to a conversational agent or robot, corresponding character distribution is determined. For example, if we specify an *extrovert* personality for ERICA, we can simulate the perception of engagement of extroverted people.

In order to use the engagement recognition model in live spoken dialogue systems, it is needed to detect listener behaviors in real time. We examined how to detect the listener behaviors with deep learning approaches [8]. Backchannels and laughing were detected from an audio signal using bi-directional long short-term memory with connectionist temporal classification (BLSTM-CTC). Head nodding was detected from a visual signal of the Kinect v2 sensor with a simpler LSTM model. Eye gaze behavior was detected also by the Kinect v2 sensor with a heuristic rule. Results of these automatic detection were used as the input to the engagement recognition model. We confirmed that the accuracy of engagement recognition was not so degraded (70.0%) even with the automatic detection of the listener behav-

Fig. 3 Real-time engagement recognition

iors. Finally, we implemented the real-time engagement recognition in the system of ERICA as shown in Fig. 3. In this study, we utilize the result of engagement recognition for generation of adaptive behaviors of ERICA.

3 Engagement-Based Adaptive Behaviors for Laboratory Guide

We implement a spoken dialogue system of ERICA where ERICA plays the role of the laboratory guide, utilizing the real-time engagement recognition model.[1] The dialogue contents of the laboratory guide are hand-crafted and consist of several research topics. A structure of the dialogue on each topic is illustrated in Fig. 4. Each topic consists of 6 subtopics where each subtopic corresponds to each research theme. Each subtopic includes two of ERICA's turns. In each turn, an engagement score is measured by the real-time recognition model, and the result is regarded as a binary: high or low, with a threshold of 0.5 for the posterior probability of engaged label. After ERICA's two turns, ERICA generates feedback utterances according to the combination of the two engagement scores in the same subtopic. We define three kinds of feedbacks as follows.

- **KEEP feedbacks**:
  When both scores are high or the scores change from low to high (increase), ERICA says feedbacks like "*You seems to be interested in my talk. I am delighted.*" in order to keep the current engagement.
- **ATTENTION feedbacks**:
  When the engagement score changes from high to low (decrease), ERICA says a different type of feedbacks such as "*Are you tired? I will explain it in easier words.*" to gain attention from the user.
- **ICE-BREAK feedbacks**:
  When both scores are low, ERICA says another type of feedbacks such as "*Are you nervous? Please relax.*" to ease the tension of the user like ice-breaking. ERICA also says like "*It would be easy to explain if you show a reaction.*" to implicitly tell a user that ERICA is monitoring their listener behaviors.

It is expected that these feedbacks make the user more engaged in the laboratory guide. In the case where a KEEP feedback is triggered, ERICA introduces an additional content of the current topic. This additional content would be beneficial for users who are potentially interested in the content of the laboratory guide. For users who are not engaged, this additional content would be difficult to understand and makes these users more disengaged. Therefore, engagement recognition needs to be accurate for precise and effective information providing.

[1] Demo video (in Japanese language) is available at https://youtu.be/53I3lhJ6aUw.

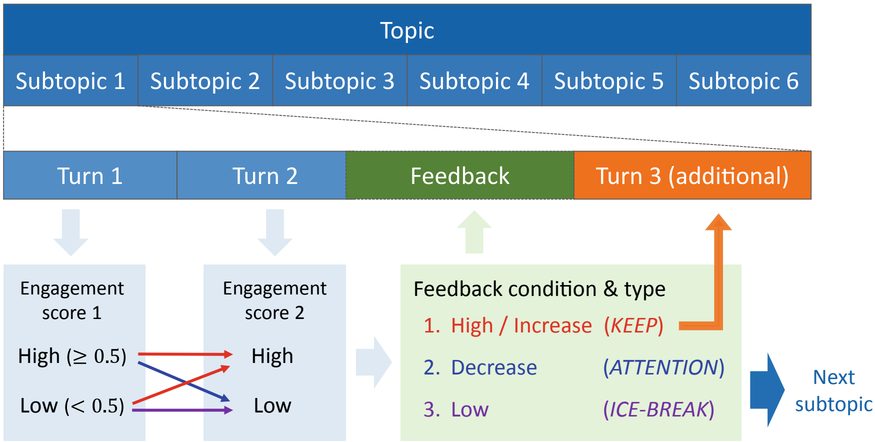


Fig. 4 Dialogue structure of laboratory guide and engagement-based adaptive behaviors

4 User Experiment

A human subject experiment was conducted in order to confirm the effectiveness of engagement recognition and also the engagement-based adaptive behaviors. Figure 5 is the snapshot of the dialogue experiment. For speech processing, we used a 16-channel microphone array to localize sound sources and to enhance the user's speech [11]. We also used end-to-end acoustic-to-word automatic speech recognition [22]. In this experiment, a subject and ERICA sat on chairs to face each other. To elicit listener behaviors of the subjects, ERICA generated backchannels and head nodding automatically during the subjects' turns [13]. The subjects were 11 persons (6 males and 5 females) who were recruited in our university. They are all native Japanese speakers. The experiment procedure is as follows. At first, each subject had a practice dialogue to get used to talking with ERICA. They practiced a simple interaction consisting of several turns. After this, ERICA explained about two research topics: automatic speech recognition and spoken dialogue systems. During the explanation, ERICA sometime gave questions toward the subject. The order of the dialogue topics was randomized among the subjects. Two experiment conditions were prepared: *engagement* and *control*. In the *engagement* condition, ERICA measured engagement scores and generated adaptive behaviors mentioned above. In the *control* condition, ERICA did not generate any adaptive behaviors, which meant only the turn 1 and 2 were explained. The dialogue of the first topic was conducted with the *control* condition, and the second topic was done with the *engagement* condition. The order of these conditions was not randomized because it was thought that the engagement-based adaptive behaviors in the *engagement* condition would affect the subjects' behaviors and impressions in the subsequent topic. Among the two conditions, measured engagement scores were compared.

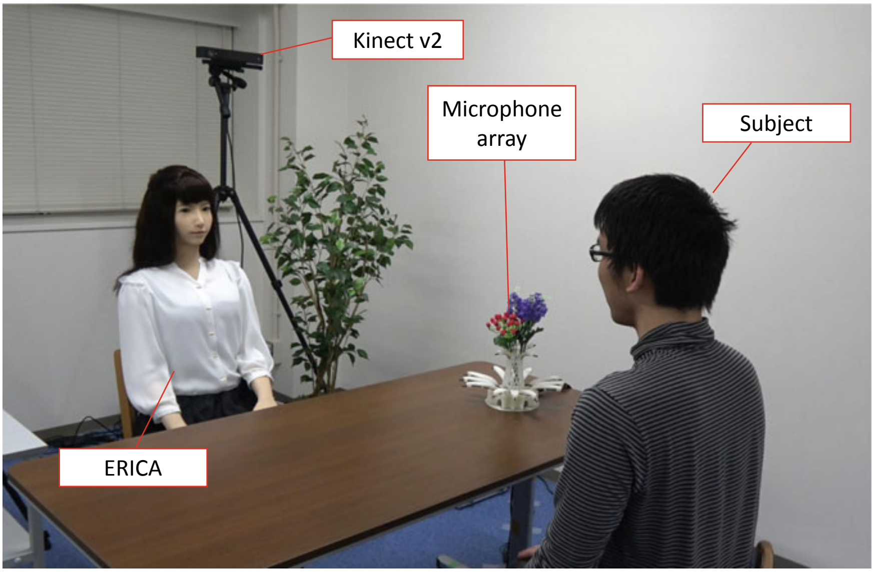


Fig. 5 Snapshot of dialogue experiment

Additionally, we also investigated subjective evaluations of users after they talked about each topic. To measure subjective evaluations, we used more specific concepts related to engagement by referring to previous works [4]. We selected related concepts as *interest* [24, 25], *continuing* [16], *willingness* [23], *rapport* [5], and *empathy* [6]. We designed question items for each concept as listed in Table 1. Two related researchers independently validated each question item by considering both the relevance to the subordinate concept and also the correctness of the question sentence. We used the 7 point scale to evaluate each item because we observed that evaluation scores tended to high and dense in a preliminary experiment. Finally, the evaluated scores were averaged for each subordinate concept. We hypothesized that the scores of the subordinate concepts would be improved in the *engagement* condition.

Average engagement scores are reported in Fig. 6. Since each topic consisted of 6 subtopics and each subtopic included two turns, the total number of turns was 12. Note that scores of the additional turns in the *engagement* condition are not included in this result. A t-test was conducted between the two conditions on all engagement scores except those of the first and second turns which were before the first-time feedback utterance. As a result, it turned out that the *engagement* condition significantly increased the engagement scores ($p = 8.06 \times 10^{-4}$). The difference between the two conditions was observed in the latter part of the dialogue. This result suggests that the adaptive behaviors in the *engagement* condition made the subjects more engaged in the dialogue. Accordingly, engagement recognition is the important function in social human-robot dialogue such as the laboratory guide.

Table 1 Evaluation scale for other concepts related to engagement (* represents invert scale.)

Concept	Question
Interest	(1) I felt the dialogue was boring. *
	(2) I wanted to listen to other topics more.
	(3) I was fascinated by the dialogue content.
	(4) I was not concerned with the dialogue content. *
Continuing	(5) I wanted to quit the dialogue during that. *
	(6) I think the dialogue should finish earlier. *
	(7) I wanted to continue the dialogue more.
Willingness	(8) I wanted to make the dialogue fulfilling.
	(9) I could not feel like talking. *
	(10) I participated in the dialogue by concentrating on that.
	(11) I felt that what I needed to do was just being there. *
	(12) I actively answered the questions from the robot.
	(13) I actively responded to the robot talk.
Rapport	(14) I liked the robot.
	(15) I felt the robot was friendly.
	(16) I was relieved when I was having the dialogue with the robot.
	(17) I could trust the dialogue content the robot talked.
Empathy	(18) I could understand what emotion the robot was having.
	(19) I could agree with the idea the robot had.
	(20) I could advance the dialogue by considering the viewpoint from the robot.
	(21) I could understand the reason why the robot had that kind of emotion.

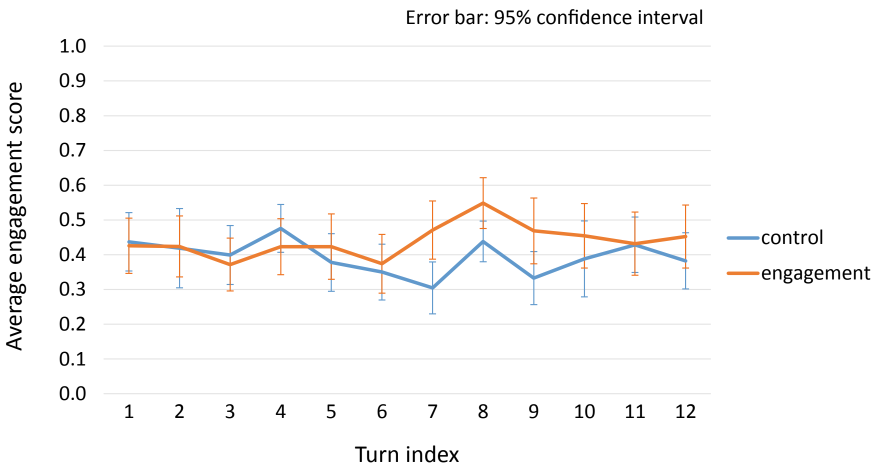


Fig. 6 Engagement scores during dialogue

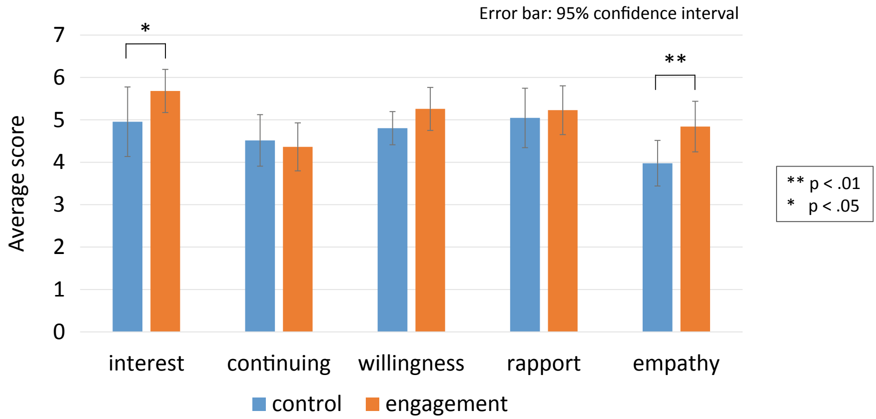


Fig. 7 Subjective evaluation on other concepts related to engagement

Average subjective scores on the related concepts are reported in Fig. 7. For each concept, a paired t-test was conducted between the two conditions. The results show that the scores of *interest* and *empathy* significantly increased in the *engagement* condition ($p < .05$ for *interest* and $p < .01$ for *empathy*). One possible reason is that the additional explanation made the subjects more interested in the research topic. Besides, the feedback responses were perceived as emotional expressions of ERICA so that they perceived higher empathy scores.

5 Conclusions

We have addressed applications of engagement recognition in order to realize social dialogue with autonomous android ERICA. In this study, the real-time engagement recognition model was applied to the dialogue of laboratory guide where ERICA plays the role of the guide. In the laboratory guide, since ERICA talks most of the time, ERICA needs to track user engagement based on listener behaviors while ERICA is speaking. ERICA was implemented to adaptively generate feedback utterances and additional explanations by measuring user engagement. The experimental results showed that the adaptive behaviors increased both the measured engagement scores and subjective evaluations of interest and empathy. Although the adaptive behaviors of ERICA were handcrafted in the current study, we will investigate how to obtain the adaptive behaviors from dialogue data in the manner of machine learning in future work.

Acknowledgements This work was supported by JST ERATO Ishiguro Symbiotic Human-Robot Interaction program (Grant Number JPMJER1401), Japan.

References

1. Castellano G, Pereira A, Leite I, Paiva A, McOwan PW (2009) Detecting user engagement with a robot companion using task and social interaction-based features. In: ICMI, pp 119–126
2. Chiba Y, Nose T, Ito A (2017) Analysis of efficient multimodal features for estimating user's willingness to talk: comparison of human-machine and human-human dialog. In: APSIPA ASC
3. DeVault D, Artstein R, Benn G, Dey T, Fast E, Gainer A, Georgila K, Gratch J, Hartholt A, Lhommet M, Lucas G, Marsella S, Morbini F, Nazarian A, Scherer S, Stratou G, Suri A, Traum D, Wood R, Xu Y, Rizzo A, Morency LP (2014) SimSensei kiosk: a virtual human interviewer for healthcare decision support. In: AAMAS, pp 1061–1068
4. Glas N, Pelachaud C (2015) Definitions of engagement in human-agent interaction. In: International workshop on engagement in human computer interaction, pp 944–949
5. Gratch J, Wang N, Gerten J, Fast E, Duffy R (2007) Creating rapport with virtual agents. In: IVA, pp 125–138
6. Hall L, Woods S, Aylett R, Newall L, Paiva A (2005) Achieving empathic engagement through affective interaction with synthetic characters. In: ICACII, pp 731–738
7. Inoue K, Hara K, Lala D, Nakamura S, Takanashi K, Kawahara T (2019) A job interview dialogue system with autonomous android ERICA. In: IWSDS. Submitted
8. Inoue K, Lala D, Takanashi K, Kawahara T (2018) Engagement recognition by a latent character model based on multimodal listener behaviors in spoken dialogue. APSIPA Trans Signal Inf Process 7(e9):1–16
9. Inoue K, Lala D, Takanashi K, Kawahara T (2018) Engagement recognition in spoken dialogue via neural network by aggregating different annotators' models. In: Interspeech, pp 616–620
10. Inoue K, Lala D, Takanashi K, Kawahara T (2018) Latent character model for engagement recognition based on multimodal behaviors. In: IWSDS
11. Inoue K, Milhorat P, Lala D, Zhao T, Kawahara T (2016) Talking with ERICA, an autonomous android. In: SIGDIAL, pp 212–215
12. Kawahara T (2018) Spoken dialogue system for a human-like conversational robot ERICA. In: IWSDS
13. Kawahara T, Yamaguchi T, Inoue K, Takanashi K, Ward NG (2016) Prediction and generation of backchannel form for attentive listening systems. In: Interspeech, pp 2890–2894
14. Lala D, Milhorat P, Inoue K, Ishida M, Takanashi K, Kawahara T (2017) Attentive listening system with backchanneling, response generation and flexible turn-taking. In: SIGDIAL, pp 127–136
15. Nakano YI, Ishii R (2010) Estimating user's engagement from eye-gaze behaviors in human-agent conversations. In: IUI, pp 139–148
16. Poggi I (2007) Mind, hands, face and body: a goal and belief view of multimodal communication. Berlin, Weidler
17. Rich C, Ponsler B, Holroyd A, Sidner CL (2010) Recognizing engagement in human-robot interaction. In: HRI, pp 375–382
18. Sanghvi J, Castellano G, Leite I, Pereira A, McOwan PW, Paiva A (2011) Automatic analysis of affective postures and body motion to detect engagement with a game companion. In: HRI, pp 305–311
19. Sidner CL, Lee C, Kidd CD, Lesh N, Rich C (2005) Explorations in engagement for humans and robots. Artif Intell 166(1–2):140–164
20. Sun, M., Zhao, Z., Ma, X.: Sensing and handling engagement dynamics in human-robot interaction involving peripheral computing devices. In: CHI, pp. 556–567 (2017)
21. Swartout W, Traum D, Artstein R, Noren D, Debevec P, Bronnenkant K, Williams J, Leuski A, Narayanan S, Piepol D, Lane C, Morie J, Aggarwal P, Liewer M, Yuan-Jen C, Gerten J, Chu S, White K (2010) Ada and grace: toward realistic and engaging virtual museum guides. In: IVA, pp 286–300
22. Ueno S, Moriya T, Mimura M, Sakai S, Shinohara Y, Yamaguchi Y, Aono Y, Kawahara T (2018) Encoder transfer for attention-based acoustic-to-word speech recognition. In: Interspeech, pp 2424–2428

23. Xu Q, Li L, Wang G (2013) Designing engagement-aware agents for multiparty conversations. In: CHI, pp 2233–2242
24. Yu C, Aoki PM, Woodruff A (2004) Detecting user engagement in everyday conversations. In: ICSLP, pp 1329–1332
25. Yu Z, Nicolich-Henkin L, Black AW, Rudnicky AI (2016) A Wizard-of-Oz study on a non-task-oriented dialog systems that reacts to user engagement. In: SIGDIAL, pp 55–63
26. Yu Z, Ramanarayanan V, Lange P, Suendermann-Oeft D (2017) An open-source dialog system with real-time engagement tracking for job interview training applications. In: IWSDS

Spoken Dialogue Robot for Watching Daily Life of Elderly People

Koichiro Yoshino, Yukitoshi Murase, Nurul Lubis, Kyoshiro Sugiyama, Hiroki Tanaka, Sakti Sakriani, Shinnosuke Takamichi, and Satoshi Nakamura

Abstract The number of aged people is increasing. The influence of solitude on both physical and mental health of those seniors is a social problem that needs an urgent solution in advanced societies. We propose a spoken dialogue robot that looks over elderly people through conversations by using functions of life-support via information navigation, attentive listening, and anomaly detection. In this paper, we describe a demonstration system implemented in the conversational robot.

1 Introduction

Social relationships are essential for promoting mental health. People who perceived the absence of positive social relationship tend to have a higher risk on both the physical and the mental health than people who have a family living together. They are depressed or socially isolated since they tend to have less communication. In the aged society, the number of aged people in solitude increases, since friends and relatives often do not have much time to be a constant companion. On the other hand, the demand for professional caregivers for older adults already outstrips supply. Thus technologies that increase opportunities for conversation are highly expected [1–4].

Communication robots are expected to solve these problems, as agents that use communicative functions to prevent isolating aged people. These systems reduce

K. Yoshino (✉) · Y. Murase · N. Lubis · K. Sugiyama · H. Tanaka · S. Sakriani · S. Nakamura
Nara Institute of Science and Technology, Nara, Japan
e-mail: koichiro@is.naist.jp

K. Yoshino · S. Sakriani · S. Nakamura
RIKEN Center for Advanced Intelligence Project, Tokyo, Japan

K. Yoshino
PRESTO, Japan Science and Technology Agency, Kawaguchi, Japan

S. Takamichi
The University of Tokyo, Tokyo, Japan

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_12

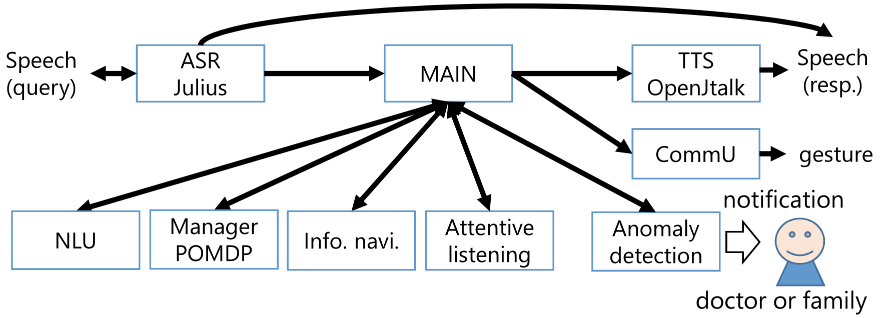


Fig. 1 System architecture

loneliness by listening to their self-disclosure attentively and increase the number of contacts to societies by providing information about daily events. These systems can also detect anomalies in aged people through daily conversations, for example, response delay caused by dementia [15].

In this work, we built a dialogue robot for watching aged people through daily conversations. The system has functions for preventing aged people from being socially isolated. The system is implemented in communicative robot called "CommU[1]" with a speech interface. The system has functions of information navigation, attentive listening, and anomaly detection, as shown in Fig. 1.

Some existing works built dialogue systems on robots for social conversations [8, 13]. These works tried to build systems in typical social roles to realize natural conversations. On the other hand, some works of dialogue systems tried to build dialogue functions for helping aged society [5, 12, 17]. Our work builds a dialogue robot in some social roles with aged people according to acquired knowledge of these existing works. Our system also tries the detection of anomaly caused by dementia.

2 System Architecture

2.1 Dialogue Scenario

The system has two dialogue functions: "information navigation" and "attentive listening." The system starts a dialogue with the information navigation function; the function initiates dialogue and informs the user recent news written in web articles in the manner of an information-navigation dialogue [19]. The system moves to attentive listening mode that listening to the user opinion on the current topic, when the focus detection predicts higher user's interest. The focus detection module is implemented by conditional random fields (CRF). The system continues to predict

[1] https://www.vstone.co.jp/products/commu/index.html.

the user's interest during the attentive listening mode by using focus detection and dialogue act classification modules based on logistic regression [18]. Once the system predicts that the user wants to know other news, the system proposes another piece of news of the day by using the information navigation module again. Before the topic transition, the system inserts a question generated by the anomaly detection module. The answer to the question is used to detect anomalies, particularly dementia in our case.

2.2 Dialogue Data

We collected dialogue data of human-human dialogue following our dialogue scenario of information navigation and attentive listening [20]. A dialogue was conducted between an aged person and a person who listened the aged person, that is, a professional counselor, a professional care-taker, or a student. Sixty dialogues were collected in total from 24 aged people, who are more than 65 years old, and 15 listeners. Each utterance was transcribed and annotated with dialogue acts to use in training the dialogue modules as follows.

2.3 Speech Interfaces

The dialogue system has speech interfaces including automatic speech recognition (ASR) and text to speech (TTS). We used Julius[2] [9] as an ASR decoder by adapting its language model with the collected dialogue data. Julius works as a server mode in deep neural network-hidden Markov model (DNN-HMM) mode and controls any speech activities of the user with its voice activity detection (VAD).

Open JTalk[3] [7] was used to generate the speech output of the system. A model of Open JTalk was first trained from the reading-style voice [14] and then adapted using the voices of a professional counselor, who is working in the area of care-taking for aged people, to build synthetic sounds that are easy for aged people to catch.

2.4 Natural Language Understanding

The user utterances transcribed by the ASR module are sent to the natural language understanding (NLU) module to extract information, which is necessary for deciding the next action of the system. Our NLU module consists of two functions: focus detection and dialogue act classification. The focus detection module detects whether

[2] http://julius.osdn.jp/.

[3] http://open-jtalk.sourceforge.net/.

an utterance made by the user contains any focus words, "an attentional state contains information about the objects, properties, relations, and discourse intentions that are most salient at any given point" [6, 18], or not, by using CRF. The dialogue act classifier selects a dialogue act for the user utterance, that is, what the intention is of the user in the dialogue level, based on logistic regression. Defined classes are based on our previous work [18].

2.5 Dialog Manager

Once the NLU module predicts the focus and the dialogue act of the user utterance, these states are used to decide the next system actions. While the system is working on the "information navigation" function, the system action is decided by a policy function trained under a partially observable Markov decision process (POMDP). The policy decides the actions of the system from three actions: topic presentation (topic changing), storytelling and question answering. The policy is trained on annotated training data and it decides an action using information form NLU modules.

When the system changes the topic, the system randomly generates a question that is used to detect anomalies. Our system randomly selects a question from a question set for predicting dementia [15, 16]. The answer of the aged person is sent to the anomaly detection server, and the result is sent to people who monitor the elderly, e.g., medical doctor or family.

2.6 Information Navigation Module

Utterances generated in information navigation mode are based on our previous study [19]. The system can provide information written in web news text (news of the day) with the following functions.

- Topic presentation: Description of available news (topics) including topic changes.
- Storytelling: Description of a summary of the current topic. The description is created from headers of web news articles.
- Question answering: Answering user's question.
- Proactive presentation: Additional information provision to previous system utterance. The sentence is generated from related news articles to the current article (topic).

We expect this module to talk with the user on a variety of domains, to prevent the isolation from the society.

2.7 Attentive Listening Module

Attentive listening mode is implemented to elicit positive emotions from the user by following our previous work [10, 11]. The system selects a response from a response pool, which is constructed from the training dialogue data. Our dialogue data is annotated with dialogue acts; thus, utterances annotated as feedback functional utterances, for example, short feedback utterances such as "I see" or "Right", are stored in the response pool. These utterances are also annotated with emotional impacts, that is, the degree of the effect on changing the emotion of the dialogue partner. The system select a response from the response pool by considering the connection to the contexts and the emotional impact.

2.8 Anomaly Detection

For the anomaly detection function, we focused on dementia. Predicting dementia from old response of old people is highly demanded on the aged society. It is crucial to predict dementia in the early stages because dementia is progressive. Dementia patients show signs in their communication even if the problem is at an early stage; however, the discovery of such symptoms from aged people in solitude is often delayed because they have fewer interactions with others than aged people living with families or other people.

Recently, there have been pieces of research on predicting early-stage dementia through human-agent interactions by asking questions [15, 16]. These works use computer avatar with spoken dialog functionalities that produces spoken queries, and try to predict dementia tendency using language and speech features extracted from user utterances by using support vector machines or logistic regressions. We implemented the dementia detection module by inserting typical questions randomly during topic transition in dialogues because it is expected that dementia can be predicted through natural conversations. Responses to these questions from aged people are sent to the dementia detection module and used for detecting. If the system predicts any indications of dementia, it will notify medical doctors or families.

2.9 Communication Robot

We implemented our system in a communication robot, CommU, since it has high expressiveness and familiarity. We set action patterns for the dialogue functions to encourage positive dialogue. These patterns were manually designed.

Acknowledgements This work is supported by JST PRESTO (JPMJPR165B) and JST CREST (JPMJCR1513).

References

1. Bemelmans R, Gelderblom GJ, Jonker P, De Witte L (2012) Socially assistive robots in elderly care: a systematic review into effects and effectiveness. J Am Med Dir Assoc 13(2):114–120
2. Broekens J, Heerink M, Rosendal H et al (2009) Assistive social robots in elderly care: a review. Gerontechnology 8(2):94–103
3. Callejas Z, López-Cózar R (2009) Designing smart home interfaces for the elderly. SIGACCESS Access Comput 95:10–16
4. Feil-Seifer D, Mataric MJ (2005) Defining socially assistive robotics. In: 9th international conference on rehabilitation robotics, 2005 (ICORR 2005). IEEE, pp 465–468
5. Georgila K, Wolters M, Moore JD, Logie RH (2010) The match corpus: a corpus of older and younger users' interactions with spoken dialogue systems. Lang Resour Eval 44(3):221–261
6. Grosz BJ, Sidner CL (1986) Attention, intentions, and the structure of discourse. Comput Linguist 12(3):175–204
7. Group HW, et al (2015) The Japanese TTS system. Open JTalk
8. Jokinen K (2018) Dialogue models for socially intelligent robots. In: International conference on social robotics. Springer, pp 127–138
9. Lee A, Kawahara T, Shikano K (2001) Julius—an open source real-time large vocabulary recognition engine. In: 7th European conference on speech communication and technology
10. Lubis N, Sakti S, Yoshino K, Nakamura S (2018) Eliciting positive emotion through affect-sensitive dialogue response generation: a neural network approach. In: AAAI
11. Lubis N, Sakti S, Yoshino K, Nakamura S (2019) Eliciting positive emotional impact in dialogue response selection. Advanced social interaction with agents. Springer, Berlin, pp 135–148
12. Miehle J, Bagci I, Minker W, Ultes S (2019) A social companion and conversational partner for the elderly. Advanced social interaction with agents. Springer, Berlin, pp 103–109
13. Milhorat P, Lala D, Inoue K, Zhao T, Ishida M, Takanashi K, Nakamura S, Kawahara T (2019) A conversational dialogue manager for the humanoid robot erica. Advanced social interaction with agents. Springer, Berlin, pp 119–131
14. Sonobe R, Takamichi S, Saruwatari H (2017) Jsut corpus: free large-scale Japanese speech corpus for end-to-end speech synthesis. arXiv preprint arXiv:1711.00354
15. Tanaka H, Adachi H, Ukita N, Ikeda M, Kazui H, Kudo T, Nakamura S (2017) Detecting dementia through interactive computer avatars. IEEE J Transl Eng Health Med 5:1–11
16. Ujiro T, Tanaka H, Adachi H, Kazui H, Ikeda M, Kudo T, Nakamura S (2018) Detection of dementia from responses to atypical questions asked by embodied conversational agents. Proc Interspeech 2018:1691–1695
17. Wolters M, Georgila K, Moore JD, MacPherson SE (2009) Being old doesn't mean acting old: how older users interact with spoken dialog systems. ACM Trans Access Comput 2(1), 2:1–2:39
18. Yoshino K, Kawahara T (2014) Information navigation system based on POMDP that tracks user focus. In: Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL), pp 32–40
19. Yoshino K, Kawahara T (2015) Conversational system for information navigation based on POMDP with user focus tracking. Comput Speech Lang 34(1):275–291
20. Yoshino K, Tanaka H, Sugiyama K, Kondo M, Nakamura S (2018) Japanese dialogue corpus of information navigation and attentive listening annotated with extended iso-24617-2 dialogue act tags. In: LREC

How to Address Humans: System Barge-In in Multi-user HRI

Nicolas Wagner, Matthias Kraus, Niklas Rach, and Wolfgang Minker

Abstract This work investigates different barge-in strategies in the context of multi-user Spoken Dialogue Systems. We conduct a pilot user study in which different approaches are compared in view of how they are perceived by the user. For this setting, the Nao robot serves as a demonstrator for the underlying multi-user Dialogue System which interrupts an ongoing conversation between two humans in order to introduce additional information. The results show that the use of supplemental sounds at this task does not necessarily lead to a positive assessment and may influence the user's perception of the reliability, competence and understandability of the system.

1 Introduction

An important goal of Spoken Dialogue Systems *(SDS)* is to communicate with users in the most efficient and intuitive way possible. Since the expectations and preferences of humans differ, it does not seem to be sufficient for a system to simply imitate human-like behaviour [2]. Instead, the implementation of exclusive social skills is required in order to increase the naturalness and flexibility of *SDS*. Moreover, an essential part of human-human interaction are conversations conducted with more than two participants. Therefore, the research on multi-user dialogue management

N. Wagner (✉) · M. Kraus · N. Rach · W. Minker
Institute of Communications Engineering, Ulm University, Ulm, Germany
e-mail: nicolas.wagner@uni-ulm.de

M. Kraus
e-mail: matthias.kraus@uni-ulm.de

N. Rach
e-mail: niklas.rach@uni-ulm.de

W. Minker
e-mail: wolfgang.minker@uni-ulm.de

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_13

focuses on scenarios where several users interact with a system simultaneously. However, due to the challenging requirements (especially in comparison to a basic human-machine interface [3]) there is currently no day-to-day system available that is capable of mastering all these tasks. Within this work, we investigate the aspect of barge-in strategies in a multi-user setup which is essential in order to build a suchlike system.

In order to obtain a physical representation in a round of talks, the employment of robots seems to be a reasonable approach to substitute a human speaker and is thus utilized in this work. The demands of social skills on a robot correspond to the complexity of its task [1], however. Related work so far concentrates on the question if a group member addresses the system or another human [8], whereas we focus on the inverse scenario and address the question of how the system should address humans when interrupting a conversation.

As stated in [6], an interruption of a human by a machine is not always unwelcome and could be useful for overcoming a problem. Nevertheless, depending on the respective task, the system needs different strategies for interrupting its users. Especially in view of naturalness and flexibility, we need to add a proactive component to avoid a static and inappropriate behaviour.

In the present work we conduct a user study in order to get an insight into user perceptions and preferences that are required to tackle these multi-user and proactive challenges. More precisely, we apply different barge-in strategies to interrupt a human-human conversation and investigate how they are perceived by the user. The respective multi-modal Dialogue System is represented by a Nao robot and the study is based on the 'Wizard-of-Oz' principle since currently no suitable *SDS* is available, as discussed above. Our results indicate that the use of different supplemental sounds has an impact on system characteristics such as reliability and competence. The remainder of this paper is as follows: Sect. 2 describes the design of the study. In the following Sect. 3, we present the outcomes and evaluate our results. Subsequently, this work closes with a conclusion and an outlook on future work.

2 Study Design

For our purpose, a configuration consisting of two human dialogue participants and one *SDS* is deemed the most intuitive and suitable for a multi-user conversation. We have chosen the humanoid robot Aldebaran Nao produced by the SoftBank Robotics Group[1] as a physical avatar for our system. Nao possesses an easy to access programming interface that allows for rapidly setting up a test environment for interaction strategies.

The human dialogue participants are an instructed supervisor, on the one hand, and a test subject on the other hand. In addition, the robot is employed as a supplementary interviewer. For guaranteeing a comparable study setup in all study conditions, we

[1] https://www.softbankrobotics.com/us/NAO.

implemented a partially determined interview as a test scenario, where both system and supervisor stick to a scripted course of questions. The subject, in contrast, is able to respond freely. This allows an untrained person to interact with the *SDS* without the need of a tutorial. The initiative changes between supervisor and robot during the interview which imitates a proactive system component. However, the robot is not allowed to engage in the conversation if a human is speaking. The participants are seated at a distance of 50 cm to each other. For this, the Nao is positioned in a sitting pose on a desk, which puts his head on the same level as the humans' heads. As an attempt to avoid technical issues, a remote operator controls the robot's actions. This type of experiment is called 'Wizard-of-Oz' because the operator is hidden from the subject during the process.

Furthermore, we select a domain about tourism to ensure a trustworthy cover story behind our setup. The scripted interaction is set to 20 turns in which the subject is asked questions about local attractions and activities. Since the questions are supposedly asked contextually by the system or supervisor, the participant believes that the robot would function autonomously. This directs the user's attention purely on the dialogue.

While the subject is convinced of the authenticity of the robot, we decide to manipulate the way the system starts to interfere in the conversation. For this study, three conditions are examined:

Direct: Baseline condition. Here, the user is addressed by the system with a direct request, e.g. "*what is your preferred leisure activity?*".

Name: In this condition the system addressed the user by adding the subject's first name to the beginning of a request, e.g. "*Alice, what is your preferred leisure activity?*".

Notification: Requests of this condition are started with a notification sound similarly to the text message notification sound of smart-phones, e.g. "**Sound* What is your preferred leisure activity?*"

The experimental environment and the surrounding conditions remain constant regardless of each test group. With these considerations we are able to state our test hypotheses:

Hypothesis 1: The addressing with the user's first name results into a more positive perception of the reliability and competence of the system. However, this has no influence on the understandability of the system.

Hypothesis 2: The use of a notification sound prior to the interruption leads to a higher distraction than the other two conditions. In addition, the perceived usefulness and the timing of the system will be assessed lower.

For our study, the questionnaires are standardized basing on the descriptions in [4, 5, 7]. The experiment starts with a welcome sequence and the providing of instructions. Subsequently, the participants are surveyed prior to and after the dialogue. We organize our questionnaire in six scales consisting of single item questions. The prior experiences of the subjects are inquired in the Pre-Trust scale (e.g. "Electrical devices cause stress."). Opposed to this, the perceived Reliability and Competence of the system is interviewed at the end of the interaction (e.g. "I can rely on the system to work properly." and "The system asks the same questions that a competent person would ask."). Additionally, the Understandability (e.g. "It is easy to understand what the system's task is."), the personal Attachment (e.g. "I perceive the system as disturbing."), as well as the Cognitive Load (e.g. "This task is challenging.") are queried afterwards. All scales are assessed with a 7-point Likert scale with a range from 1 = "totally disagree" to 7 = "totally agree". The next section shows our outcomes.

3 Evaluation

We surveyed a total of 17 student participants for our study. Unfortunately two questionnaires had to be excluded due to technical errors on the Nao robot during the experiment. The evaluation could thus be performed with five participants for each of the three test groups. We had 10 female and 5 male subjects with an average age of $M = 25.07$ ($SD = 2.43$). Despite the small number of test persons, it was possible to obtain significant results. We tested if previous experiences with technical devices of the user differed in the three test conditions. By applying the Kruskal–Wallis test, we showed that these variables had no influence on the final results (since $p = 0.061$).

For three of the remaining five question scales, the Cronbach's alpha is sufficient. To be precise, the lowest alpha of each category equals to: Reliability ($\alpha = 0.73$), Competence ($\alpha = 0.89$), Understandability ($\alpha = 0.77$), Attachment ($\alpha = 0.13$), and Cognitive Load ($\alpha = 0.03$). Due to this limitation, the last two scales could not be examined further. In those cases, however, we decided to select single questions for the following evaluation. The mean values and standard deviations of the scales with an adequate Cronbach's alpha are presented below:

Table 1 indicates that the direct approach was rated with the highest means of all three subject groups in the scales Reliability, Competence, and Understandability. In contrast, the group addressed by the user's first name gave the lowest assessment except for Understandability. Hence, the subjects appeared to perceive a system which addresses the user either by first name or by notification sound as less competent and reliable. In addition, three individual questions had been determined for evaluation: "The system is useful.", "The timing of the questions is well-chosen.", and

Table 1 Mean value and standard deviation of question scales with Cronbach's alpha >0.7

	Reliability	Competence	Understandability
Direct	5.96 0.71	6.00 0.68	4.96 1.99
Name	4.56 1.38	4.88 0.88	4.90 0.94
Notification	5.32 0.77	4.92 1.40	3.95 0.89

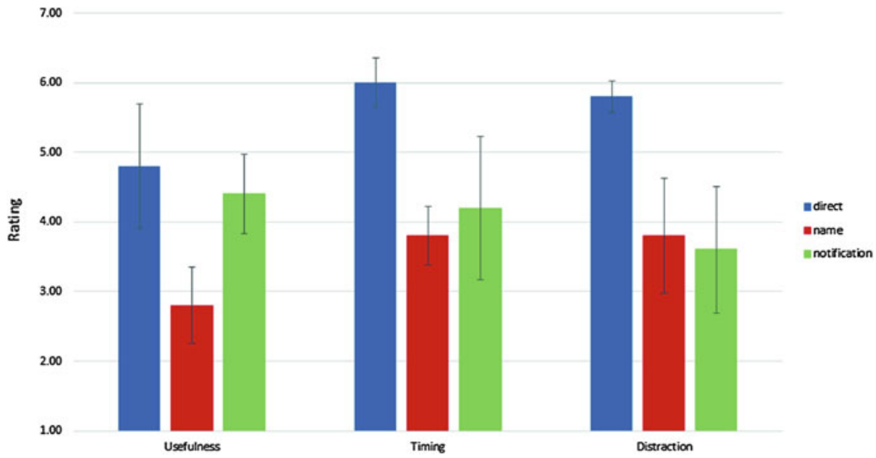


Fig. 1 Mean rating and standard deviation of the individual questions

"The system is distracting me.". The selection of these three questions was based on their importance in multi-user *SDS*. Their respective ratings are illustrated in Fig. 1. As can be seen there, the direct approach was predominantly considered positive. We utilized the Mann–Whitney-U test to identify significant differences among the test groups. Between the direct and the name group, the Timing ($p = 0.01$) and the Distraction ($p = 0.04$) were significant. Moreover, the Usefulness ($p = 0.09$), the Reliability ($p = 0.09$), and the Competence ($p = 0.06$) showed a tendency, nevertheless the small group of test subjects didn't allow for significant results. In the case of the direct versus the notification group, the Distraction ($p = 0.04$) exposed significant differences. All other scales showed no significant results in the Mann–Whitney-U test. Between the name and the notification group, the Usefulness ($p = 0.05$) is significant while all other scales have p-values $p \gg 0.14$.

According to this, the first hypothesis has been partially disproved. The addressing with the user's first name resulted in a poorer rating of reliability and competence. Our assumption of a constant understandability was correct. Furthermore, the second hypothesis was not confirmed by our evaluation. Based on the adequate values for Cronbach's alpha and the Mann–Whitney-U tests, we were nevertheless able to obtain several statistically meaningful results. The following section summarizes this work in a conclusion and gives an outlook on future work.

4 Conclusion and Future Work

This paper presented the results of a pilot study with 15 participants. We described our experimental design in a tourism domain where the subjects were alternately addressed by a human supervisor and a robotic assistant. It was investigated which approach is the smartest for a system to barge-in in a human-human interaction. For this, the scenario was specified in such a way that the interruptions by the robot occur in different styles. The results of our experiment showed that using supplemental sounds does not improve the user's perception of the capabilities of the system. In fact, especially the subject's first name has been experienced as an unpleasant start for a system question. This could be due to the fact that the addressing style remained the same throughout the dialogue, which could be annoying for participants. For future studies, we plan to extend the possibilities of the system to start an interaction. We intend to conduct similar experiments soon.

However, a higher number of subjects would allow for statistically more reliable predictions. We are also aware that the use of proactivity requires further investigation. The application of statistical methods seems to be inevitable in this context. On the other hand, this work has brought us important insights into multi-user Dialogue Systems with robots which can be used for subsequent experiments. In order to enable a natural and flexible interaction, future participants should not only be forced to answer questions, but also be given the option to ask the system themselves.

Acknowledgements This work is part of the "RobotKoop" project that is funded by the Bundesministerium für Bildung und Forschung (BMBF, German Federal Ministry of Education and Research).

References

1. Dautenhahn K (2007) Socially intelligent robots: dimensions of human-robot interaction. Philos Trans R Soc Lond B: Biol Sci 362(1480):679–704
2. Edlund J, Gustafson J, Heldner M, Hjalmarsson A (2008) Towards human-like spoken dialogue systems. Speech Commun 50(8–9):630–645
3. Foster ME, Gaschler A, Giuliani M, Isard A, Pateraki M, Petrick R (2012) Two people walk into a bar: Dynamic multi-party social interaction with a robot agent. In: Proceedings of the 14th ACM international conference on multimodal interaction. ACM, pp 3–10
4. Karrer K, Glaser C, Clemens C, Bruder C (2009) Technikaffinität erfassen-der fragebogen ta-eg. Der Mensch im Mittelpunkt technischer Systeme 8:196–201
5. Klepsch M, Schmitz F, Seufert T (2017) Development and validation of two instruments measuring intrinsic, extraneous, and germane cognitive load. Front Psychol 8:1997. https://doi.org/10.3389/fpsyg.2017.01997. https://www.frontiersin.org/article/10.3389/fpsyg.2017.01997
6. McFarlane DC, Latorella KA (2002) The scope and importance of human interruption in human-computer interaction design. Hum-Comput Interact 17(1):1–61
7. Merritt SM, Heimbaugh H, LaChapell J, Lee D (2013) I trust it, but i don't know why: Effects of implicit attitudes toward automation on trust in an automated system. Hum Factors 55(3):520–534
8. Richter V, Carlmeyer B, Lier F, Meyer zu Borgsen S, Schlangen D, Kummert F, Wachsmuth S, Wrede B (2016) Are you talking to me?: Improving the robustness of dialogue systems in a multi party HRI scenario by incorporating gaze direction and lip movement of attendees. In: Proceedings of the 4th international conference on human agent interaction. ACM, pp 43–50

Bone-Conducted Speech Enhancement Using Hierarchical Extreme Learning Machine

Tassadaq Hussain, Yu Tsao, Sabato Marco Siniscalchi, Jia-Ching Wang, Hsin-Min Wang, and Wen-Hung Liao

Abstract Deep learning-based approaches have demonstrated promising performance for speech enhancement (SE) tasks. However, these approaches generally require large quantities of training data and computational resources for model training. An alternate hierarchical extreme learning machine (HELM) model has been previously reported to perform SE and has demonstrated satisfactory results with a limited amount of training data. In this study, we investigate application of the HELM model to improve the quality and intelligibility of bone-conducted speech. Our experimental results show that the proposed HELM-based bone-conducted SE framework can effectively enhance the original bone-conducted speech and outperform a deep denoising autoencoder-based bone-conducted SE system in terms of speech quality and intelligibility with improved recognition accuracy when a limited quantity of training data is available.

T. Hussain
TIGP-SNHCC, Department of Computer Science, National Chengchi University, Taipei, Taiwan
e-mail: tass.hussain@iis.sinica.edu.tw

Y. Tsao
Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan
e-mail: yu.tsao@citi.sinica.edu.tw

S. M. Siniscalchi (✉)
Department of Computer Science, Kore University of Enna, Enna, Italy
e-mail: marco.siniscalchi@unikore.it

J.-C. Wang
Department of Computer Science and Information Engineering, National Central University, Taoyuan City, Taiwan
e-mail: jcw@csie.ncu.edu.tw

H.-M. Wang
Institute of Information Science, Academia Sinica, Taipei, Taiwan
e-mail: whm@iis.sinica.edu.tw

W.-H. Liao
Department of Computer Science, National Chengchi University, Taipei, Taiwan
e-mail: whliao@cs.nccu.edu.tw

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_14

1 Introduction

Speech enhancement (SE) refers to a technique to modify an input speech signal to a target signal with improved speech quality and intelligibility. The input speech signal is usually distorted by additive or convolutive noises or recording device constraints. In this work, we propose a hierarchical extreme learning machine learning (HELM) model to convert a speech utterance recorded with a bone-conducted microphone (BCM) to one from an air-conducted microphone (ACM). Unlike an ACM that records speech signals directly, a BCM captures speech signals based on the vibrations of the speaker's skull. The speech signals recorded with a BCM are robust against noise while some high frequency components may be missing compared to the speech signals recorded with an ACM.

A number of filtering-based and probabilistic solutions have been proposed in the past to convert BCM utterances to ACM utterances. In [12], the BCM utterances were passed through a designed reconstruction filter to improve quality. In [19] and [20], BCM and ACM utterances were combined for SE and automatic speech recognition (ASR) in non-stationary noisy environments. In [4], a probabilistic optimum filter (POF)-based algorithm was used to estimate the clean features from the combination of standard and throat microphone signals. Thang et al. [16] restored bone conducted speech in noisy environments based on a modulation transfer function (MTF) and a linear prediction (LP) model. Later, Tajiri et al. [14] proposed a noise suppression technique based on non-negative tensor factorization using a body-conducted microphone known as a nonaudible murmur (NAM) microphone.

Recently, neural network based approaches have shown tremendous performance for SE. In these approaches, non-linear mapping functions are estimated to transform the source speech utterances to target speech utterances using a learning-based model. The models are generally trained with source–target utterance pairs. Numerous neural network frameworks have been used and have demonstrated exceptional performance for speech processing. For example, in [9], the authors applied a deep denoising autoencoder (DDAE) framework by stacking multiple denoising autoencoders and demonstrated state-of-the-art performance for SE. Meanwhile, in [18], a deep neural network (DNN) was used to handle a wide range of additive noises for the SE task. In addition, a signal-to-noise ratio (SNR)-aware convolutional neural network was proposed by Fu et al. [2] for SE. Similarly, in [3], a fully convolutional network (FCN)-based architecture is used to optimize the intelligibility of speech, along with the model parameters, during SE. In [1], a bidirectional long short-term memory (BLSTM)-based framework was utilized for SE and ASR. Furthermore, generative adversarial networks have also been deployed for SE [10, 17]. More recently, the DDAE framework has been applied to BCM–ACM SE and has been noted to provide satisfactory generalization and speech recognition performance [8]. Despite notable improvements achieved by the deep learning models over the conventional approaches, deeper structure-based methods typically require large computational resources and adequate quantities of training data for effective learning.

In addition to the deep learning models that require large quantities of data to train the models and learn the mapping functions, an HELM model has been proposed to perform SE and has been confirmed to achieve very good performance with limited quantities of training data [7]. In this study, we further investigate the HELM model for enhancing BCM speech. The experimental results verify that the proposed SE framework notably improves the original BCM speech and outperforms the previous DDAE-based SE framework in terms of two standardized objective measures, namely perceptual evaluation of speech quality (PESQ) [11] and short-time objective intelligibility (STOI) [13], as well as ASR performance.

2 Proposed HELM-Based BCM Speech Enhancement System

The extreme learning machine (ELM) was proposed by Huang et al. [6] for shallow neural feed-forward architectures to address the slow gradient issue. In addition to ELM, the hierarchical structure of ELM, termed as HELM, was proposed by Tang et al. [15] to extract feature representation in a hierarchical manner. HELM consists of both unsupervised and supervised (semi-supervised) stages, in which the compact feature representations are extracted using the unsupervised ELM-based sparse autoencoders followed by the supervised regression/classification stage. Figure 1 shows the offline and online phases of the proposed HELM-based SE system. During the offline phase, the speech utterances recorded using both BCM and ACM are divided into short segments of frames and subsequently converted into the frequency domain using short-time Fourier transform (STFT). The frame-based 160

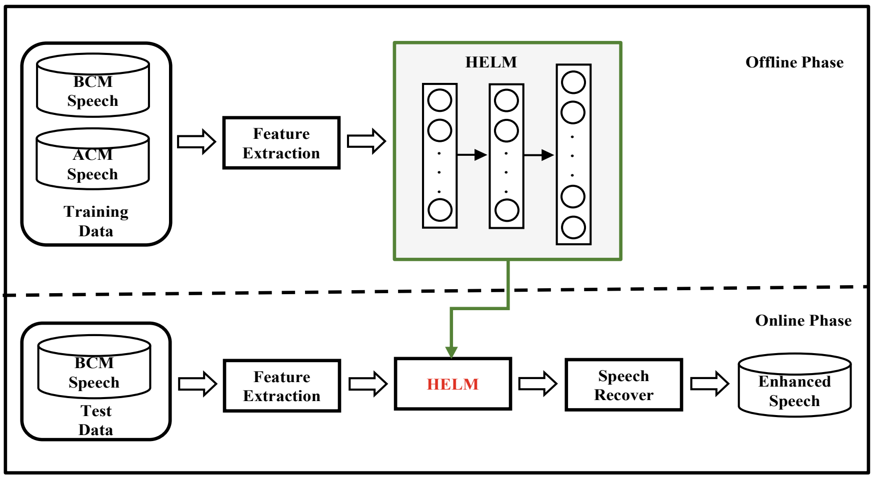


Fig. 1 HELM-based SE architecture

dimensional Mel spectral features of the BCM–ACM utterance pairs are estimated and analyzed by the HELM to learn the mapping function to convert the BCM Mel spectral features to the corresponding ACM Mel spectral features.

In the testing phase, the incoming BCM test utterance is initially segmented into short frames using STFT and subsequently converted into 160 dimensional Mel spectral features. The BCM Mel spectral features are further converted by the trained HELM to acquire the enhanced Mel spectral features. The inverse STFT is then applied to the enhanced Mel spectral features together with the phase of the original BCM utterance to obtain the restored speech. More details for the online and offline phases of the HELM framework can be found in [7].

3 Experiments and Results

3.1 Experimental Setup and Dataset Description

The dataset used in our experiments is the same as the one used in [8]. The utterances were spoken by a native speaker and recorded using ACM and BCM microphones simultaneously with the transcript of the Taiwan Mandarin hearing in noise test (TMHINT) [8]. The dataset was originally recorded using a sampling rate of 44.1 kHz and was further resampled to 16 kHz for processing. In this study, we selected 270 utterances from the complete dataset, among which 200 utterances were randomly selected and used as the training data, and the remaining 70 utterances were selected as the testing data. There was no overlap between the training and testing utterances.

We evaluated the proposed approach using two standardized objective measures: PESQ and STOI. The PESQ measures the speech quality of the estimated speech signal with the clean speech signal as a reference. The PESQ score ranges between −0.5 and 4.5, where a higher score denotes better speech quality. The STOI measures the intelligibility of the estimated speech with the clean speech signal as a reference. The STOI score ranges between 0.0 and 1.0, and a higher score indicates better speech intelligibility. In this study, we used 160 dimensional Mel spectral features along with 80,000 patches for the BCM and ACM training samples. In the experiments, we did not use the context information of the input speech vectors, i.e., no neighboring input speech vectors were considered.

3.2 HELM-Based SE System

We first utilize the performance of the proposed HELM system for converting the BCM speech to ACM speech. Table 1 demonstrates the average PESQ and STOI scores of the unprocessed BCM and HELM enhanced speech. The utterances

Table 1 Average PESQ and STOI scores of the unprocessed BCM speech and the HELM enhanced speech trained with the BCM/ACM utterance pairs

Method	PESQ	STOI
Unprocessed	2.5775	0.7254
HELM(BCM/ACM)	2.7903	0.8329

Table 2 Average PESQ and STOI scores of the unprocessed BCM speech and DDAE- and HELM-enhanced speech trained with the BCM/ACM(IE) utterance pairs

Method	PESQ	STOI
DDAE(BCM/ACM-IE)	2.7145	0.8470
HELM(BCM/ACM-IE)	2.8454	0.8306

recorded with the BCM only contain the lower frequency components of the utterances. The HELM framework was trained using the BCM/ACM training utterance pairs with small numbers of hidden neurons (200, 200, and 500), where 200, 200, and 500 are the number of hidden neurons for the first, second, and third layer of HELM, with a sigmoid activation function. The regularization parameters in HELM were the same as those used in [7].

From Table 1, it is clear that HELM improves the speech quality and intelligibility by obtaining a remarkable improvement in the PESQ and STOI scores compared to the unprocessed BCM utterances. Next, we applied a speech intelligibility index (SII)-based post-filter [8] on the utterances recorded with the ACM to train our HELM framework. That is, HELM was trained using the BCM/ACM(IE) utterance pairs, where ACM(IE) represents the ACM utterances processed by the SII post-filter. The purpose of SII-based post-filtering is to consider the critical band importance function, which correlates speech intelligibility for humans. Table 2 shows the performance comparison between the DDAE- and HELM-based enhanced utterances. For fair comparison, we employed a three-layer DDAE structure, where each layer had 300 hidden neurons ($300 \times 3 = 900$ hidden neurons), with a sigmoid activation function. The table shows that HELM outperformed the unprocessed BCM utterances and DDAE-based enhanced utterances in terms of PESQ score with a reasonable margin when the systems are trained using BCM-ACM(IE) utterances. However, DDAE provided a higher intelligibility (STOI) score as compared with the proposed HELM framework. In addition, comparing the results in Tables 1 and 2, we can see that the PESQ performance of HELM can be improved by using the BCM/ACM(IE) training utterances.

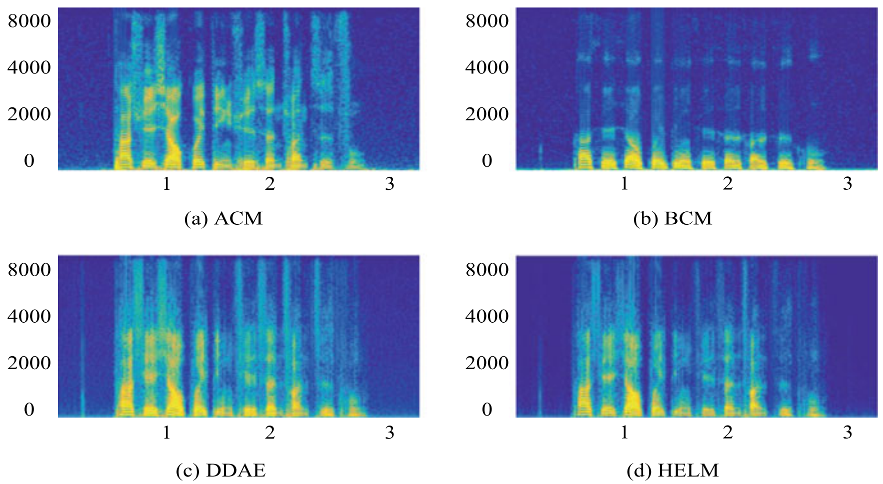


Fig. 2 Spectrograms of the enhanced test utterances using the **c** DDAE and **d** HELM of the **a** ACM and **b** BCM utterances. For each figure, the x-axis denotes the time in seconds, and the y-axis represents the frequency in Hertz

3.3 Spectrogram Analysis

In this section, we analyze the spectrogram of the converted speech signal yielded by HELM and DDAE. Figure 2a, b show the spectrograms of the BCM and corresponding ACM utterances. Figure 2c, d display the spectrograms of the DDAE- and HELM- enhanced speech. From Fig. 2c, d, we observe that both of the frameworks (DDAE and HELM) have successfully converted the BCM utterance to the enhanced utterance closer to the original ACM (Fig. 2a) utterance. Meanwhile, HELM achieves better speech quality with PESQ = 2.9633 as compared to DDAE whose PESQ = 2.6027, where the original BCM utterance has PESQ = 2.2619.

3.4 Automatic Speech Recognition

In addition to the speech quality and sound intelligibility measures, we also compared the recognition results of the speech processed by the HELM and DDAE frameworks. As discussed in the previous section, the original BCM utterances only contain the low frequency components of the speech signals, which could result in a poor recognition performance. We computed the ASR performance of the DDAE- and HELM- enhanced speech and the original BCM test utterances using Google ASR [5]. The testing results of the ACM utterances were also listed as the upper bound. Table 3 presents the character error rate (CER) evaluated on the 70 test utterances. From Table 3, we first note that BCM speech achieved higher CERs compared to

Table 3 CERs of the original ACM and BCM test utterances and DDAE- and HELM-enhanced speech

	ACM	BCM	DDAE	HELM
CER (%)	1.0	12.13	10.98	8.27

ACM speech. Next, we can see that both frameworks (DDAE and HELM) provide a lower CER compared to the unprocessed BCM speech. Moreover, HELM clearly outperforms DDAE in the ASR experiments.

3.5 Sensitivity/Stability Towards the Training Data

Next, we intend to analyze the sensitivity of the proposed HELM framework with the quantity of training data. In our previous sections, we used 80000 Mel spectral patches (MSP) of the samples, randomly selected from the training set, to train the enhancement models. In this set of experiments, we investigated the performance using different sizes for the training data from 200 to 500, 1000, 5000, 10000, 20000, 40000, and 80000 MSPs.

Figures 3 and 4 show the brief summaries of the two frameworks in terms of average PESQ and STOI, respectively. From Fig. 3, we can note that the PESQ scores of the DDAE framework has improved from 1.4004 to 1.4732 when the size of the training data increases from 200 to 500 MSP. The same trend of improvement can be seen when the size of the training data is increased from 500 to 80000 MSP for the DDAE framework. On the other hand, HELM presents less sensitivity towards the reduction in the size of the training data. The PESQ score for HELM configuration escalated from 2.1867 to 2.4605 when the size of the training data only increased

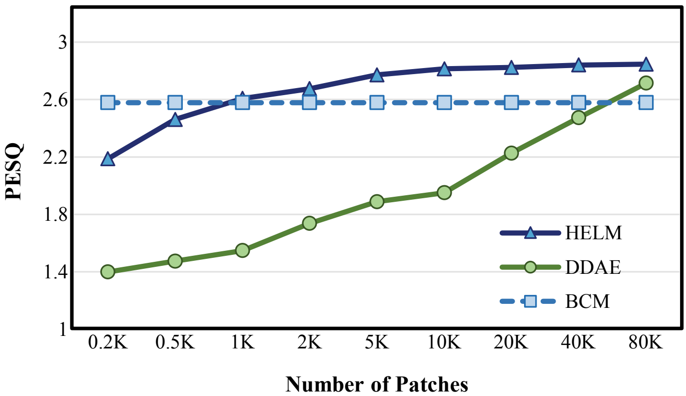


Fig. 3 Average PESQ scores for DDAE and HELM SE frameworks using different amounts of training data

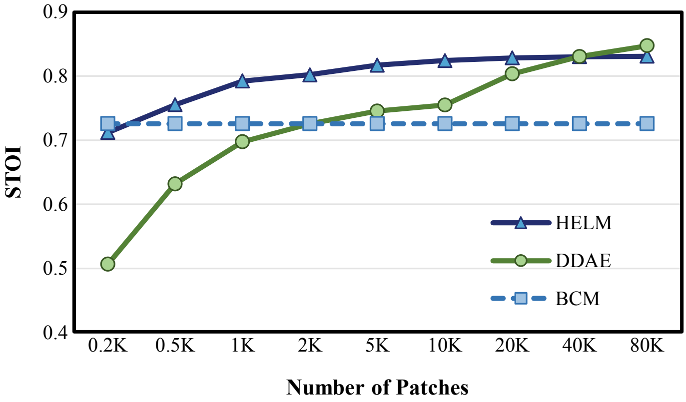


Fig. 4 Average STOI scores for DDAE and HELM SE frameworks using different amounts of training data

from 200 to 500 MSP, as shown in Fig. 3. Similarly, the PESQ score further escalated to 2.8454 (with 80000 MSP) from 2.6018 (with 1000 MSP). It is very interesting to observe that DDAE framework require more than 10000 MSP to increase the PESQ score beyond 2.0 (PESQ = 1.9517 with 10000 MSP), whereas HELM can achieve PESQ = 2.1867 even with 200 MSP.

Next from Fig. 4, the STOI score for the DDAE framework was low when the patches were 200 MSP. In contrast, HELM produced comparable STOI scores to the unprocessed BCM speech when the training data size was 200 MSP. The STOI for DDAE dropped from 0.8470 to 0.5066 while the drop was not that severe for HELM as it only declined from 0.8306 to 0.7116 when the amount of MSP was reduced from 80000 to 200. The results from Figs. 3 and 4 show that both DDAE and HELM can yield improved speech quality and intelligibility when sufficient training data are available (more than 80000MSP), while HELM gave improved performance even with 2000 MSP.

Moreover, we evaluated the ASR results of the DDAE and HELM frameworks for different amounts of training data. Figure 5 shows the impact of the number of MSPs on the CER results of the DDAE and HELM frameworks. From the figure, we can observe a similar trend for the performance of CER as that observed for PESQ and STOI. Overall, the CERs of the two frameworks are reduced when the MSP increases from 1000 to 80000. Moreover, HELM yields a consistent and stable recognition performance with small CER as compared with DDAE for different numbers of MSP. The CERs of the DDAE decreased consistently and later reduced dramatically when the MSP increased beyond 10000 MSP, indicating the impact of the training data on the performance of the DDAE framework. As compared to DDAE, HELM can yield reduced CER as compared to the BCM even with 5000 MSP. The results again confirm the advantage of HELM over the DDAE when the amount of training data is limited.

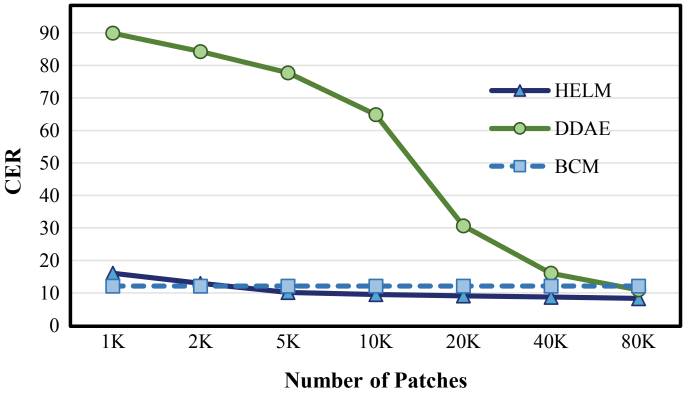


Fig. 5 CER results of DDAE and HELM frameworks using different amounts of training data

4 Conclusion

In this study, an HELM-based BCM SE approach was proposed. We evaluated the proposed approach using two standardized objective measures, i.e., PESQ and STOI. For comparison, a DDAE-based SE system was established and used to test performance. Meanwhile, the CERs were tested to compare the performance of the DDAE and HELM enhanced speech. The experimental results have confirmed the effectiveness of the proposed HELM-based SE framework by maintaining high speech quality and intelligibility with high recognition accuracy. Since deep learning based approaches generally require large quantities of training data to learn complex nonlinear relationships between the input and output, we examined the sensitivities of the DDAE and HELM frameworks with different quantities of training samples. The performance of the HELM-based SE framework proved to be consistent and provided superior performance compared with the DDAE-based SE framework. This is the first attempt that has successfully applied HELM framework for bone-conducted SE. Because of no fine-tuning or adjustment of parameters during the training phase, HELM-based models are highly suitable for applications in embedded and mobile devices.

References

1. Chen Z, Watanabe S, Erdogan H, Hershey JR (2015) Speech enhancement and recognition using multi-task learning of long short-term memory recurrent neural networks. In: Proceedings of ISCA, pp 3274–3278
2. Fu SW, Tsao Y, Lu X (2016) SNR-Aware convolutional neural network modeling for speech enhancement. In: Proceedings of INTERSPEECH, pp 3768–3772
3. Fu SW, Wang TW, Tsao Y, Lu X, Kawai H (2018) End-to-end waveform utterance enhancement for direct evaluation metrics optimization by fully convolutional neural networks. IEEE/ACM

Trans Audio Speech Lang Proces 26:1570–1584. https://doi.org/10.1109/TASLP.2018.2821903
4. Graciarena M, Franco H, Sonmez K, Bratt H (2003) Combining standard and throat microphones for robust speech recognition. IEEE Sig Process Lett 10:72–74. https://doi.org/10.1109/LSP.2003.808549
5. Google (2017) Cloud speech API. https://cloud.google.com/speech/
6. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501
7. Hussain T, Siniscalchi SM, Lee CC, Wang SS, Tsao Y, Liao WH (2017) Experimental study on extreme learning machine applications for speech enhancement. IEEE Access 5:25542–25554. https://doi.org/10.1109/ACCESS.2017.2766675
8. Liu HP, Tsao Y, Fuh CS (2018) Bone-conducted speech enhancement using deep denoising autoencoder. Speech Commun 104:106–112
9. Lu X, Tsao Y, Matsuda S, Hori C (2013) Speech enhancement based on deep denoising autoencoder. In: Proceedings of INTERSPEECH, pp 436–440
10. Pascual S, Bonafonte A, Serra J (2017) SEGAN: speech enhancement generative adversarial network. arXiv:1703.09452
11. Rix AW, Beerends JG, Hollier MP, Hekstra AP (2001) Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs. In: Proceedings of ICASSP, pp 749–752
12. Shinamura T, Tomikura T (2005) Quality improvement of bone-conducted speech. In: Proceedings of ECCTD, pp 1–4
13. Taal CH, Hendriks RC, Heusdens R, Jensen J (2010) A short-time objective intelligibility measure for time-frequency weighted noisy speech. In: Proceedings of ICASSP, pp 4214–4217
14. Tajiri Y, Kameoka H, Toda T (2017) A noise suppression method for body-conducted soft speech based on non-negative tensor factorization of air-and body-conducted signals. In: Proceedings of ICASSP, pp 4960–4964
15. Tang J, Deng C, Huang GB (2016) Extreme learning machine for multilayer perceptron. IEEE Trans Neural Netw Learn Syst 27:809–821. https://doi.org/10.1109/TNNLS.2015.2424995
16. Thang TV, Kimura K, Unoki M, Akagi M (2006) A study on restoration of bone-conducted speech with MTF-based and LP-based models. J Sig Process 10:407–417
17. Van Den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kavukcuoglu K (2016) WaveNet: A generative model for raw audio. arXiv:1609.03499
18. Xu Y, Du J, Dai LR, Lee CH (2015) A regression approach to speech enhancement based on deep neural networks. IEEE/ACM Trans Audio Speech Lang Process 23:7–19. https://doi.org/10.1109/TASLP.2014.2364452
19. Zhang Z, Liu Z, Sinclair M, Acero A, Deng L, Droppo J, Zheng Y (2004) Multi-sensory microphones for robust speech detection, enhancement, and recognition. In: Proceedings of ICASSP, pp 781–784
20. Zheng Y, Liu Z, Zhang Z, Sinclair M, Droppo J, Deng L, Huang X (2003) Air-and bone-conductive integrated microphones for robust speech detection and enhancement. In: Proceedings of ASRU, pp 249–254

Dialogue Evaluation and Analysis

Benchmarking Natural Language Understanding Services for Building Conversational Agents

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser

Abstract We have recently seen the emergence of several publicly available Natural Language Understanding (NLU) toolkits, which map user utterances to structured, but more abstract, Dialogue Act (DA) or Intent specifications, while making this process accessible to the lay developer. In this paper, we present the first wide coverage evaluation and comparison of some of the most popular NLU services, on a large, multi-domain (21 domains) dataset of 25 K user utterances that we have collected and annotated with Intent and Entity Type specifications and which will be released as part of this submission (https://github.com/xliuhw/NLU-Evaluation-Data). The results show that on Intent classification Watson significantly outperforms the other platforms, namely, Dialogflow, LUIS and Rasa; though these also perform well. Interestingly, on Entity Type recognition, Watson performs significantly worse due to its low Precision (At the time of producing the camera-ready version of this paper, we noticed the seemingly recent addition of a 'Contextual Entity' annotation tool to Watson, much like e.g. in Rasa. We'd threfore like to stress that this paper does *not* include an evaluation of this feature in Watson NLU.). Again, Dialogflow, LUIS and Rasa perform well on this task.

(Work done when Pawel was with Emotech North LTD).

X. Liu (✉) · A. Eshghi · V. Rieser
Heriot-Watt University, Edinburgh EH14 4AS, UK
e-mail: x.liu@hw.ac.uk

A. Eshghi
e-mail: a.eshghi@hw.ac.uk

V. Rieser
e-mail: v.t.rieser@hw.ac.uk

P. Swietojanski
The University of New South Wales, Sydney, Australia
e-mail: p.swietojanski@unsw.edu.au

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_15

1 Introduction

Spoken Dialogue Systems (SDS), or Conversational Agents are ever more common in home and work environments, and the market is only expected to grow. This has prompted industry and academia to create platforms for fast development of SDS, with interfaces that are designed to make this process easier and more accessible to those without expert knowledge of this multi-disciplinary research area.

One of the key SDS components for which there are now several such platforms available is the Natural Language Understanding (NLU) component, which maps individual utterances to structured, abstract representations, often called Dialogue Acts (DAs) or Intents together with their respective arguments that are usually Named Entities within the utterance. Together, the representation is taken to specify the semantic content of the utterance as a whole in a particular dialogue domain.

In the absence of reliable, third-party—and thus unbiased—evaluations of NLU toolkits, it is difficult for users (which are often conversational AI companies) to choose between these platforms. In this paper, our goal is to provide just such an evaluation: we present the first systematic, wide-coverage evaluation of some of the most commonly used[1] NLU services, namely: Rasa,[2] Watson,[3] LUIS[4] and Dialogflow.[5] The evaluation uses a new dataset of 25 k user utterances which we annotated with Intent and Named Entity specifications. The dataset, as well as our evaluation toolkit will be released for public use.

2 Related Work

To our knowledge, this is the first wide coverage comparative evaluation of NLU services—those that exist tend to lack breadth in Intent types, Entity types, and the domains studied. For example, recent blog posts [3, 5], summarise benchmarking results for 4 domains, with only 4 to 7 intents for each. The closest published work to the results presented here is by [1], who evaluate 6 NLU services in terms of their accuracy (as measured by precision, recall and F-score, as we do here) on 3 domains with 2, 4, and 7 intents and 5, 3, and 3 entities respectively. In contrast, we consider the 4 currently most commonly used NLU services on a large, new data set, which contains 21 domains of different complexities, covering 64 Intents and 54 Entity types in total. In addition, [2] describe an analysis of NLU engines in terms of their usability, language coverage, price etc., which is complimentary to the work presented here.

[1] According to anecdotal evidence from academic and start-up communities.

[2] https://rasa.com/.

[3] https://www.ibm.com/watson/ai-assistant/.

[4] https://www.luis.ai/home.

[5] https://dialogflow.com/.

3 Natural Language Understanding Services

There are several options for building the NLU component for conversational systems. NLU typically performs the following tasks: (1) Classifying the user Intent or Dialogue Act type; and (2) Recognition of Named Entities (henceforth NER) in an utterance.[6] There are currently a number of service platforms that perform (1) and (2): commercial ones, such as Google's Dialogflow (formerly Api.ai), Microsoft's LUIS, IBM's Watson Assistant (henceforth Watson), Facebook's Wit.ai, Amazon Lex, Recast.ai, Botfuel.io; and open source ones, such as Snips.ai[7] and Rasa. As mentioned above, we focus on four of these: Rasa, IBM's Watson, Microsoft's LUIS and Google's Dialogflow. In the following, we briefly summarise and discuss their various features. Table 1 provides a summary of the input/output formats for each of the platforms.

Table 1 Input requirements and output of NLU services

Service	Input (Training)	Output (Prediction)
Rasa	JSON or Markdown. Utterances with annotated intents and entities. Can provide synonym and regex features.	JSON. The intent and intent_ranking with confidence. A list of entities without scores
Dialogflow	JSON. List of all entity type names and values/synonyms. Utterance samples with annotated intents and entities. Need to specify the expected returning entities as parameters for each intent.	JSON. The intent and entities with values. Overall score returned, not specific to Intent or Entity. Other returned info related to dialogue app
LUIS	JSON, Phrase list and regex patterns as model features, hierarchical and composites entities. List of all intents and entity type names. Utterance samples with annotated intents and entities	JSON. The intent with confidence. A list of entities with scores
Watson	CSV. List of all utterances with Intent label. List of all Entities with values. No annotated entities in an utterance needed.	JSON. The intent with confidence. A list of entities and confidence for each. Other info related to dialogue app

[6]Note that, one could develop one's own system using existing libraries, e.g. sk_learn libraries http://scikit-learn.org/stable/, spaCy https://spacy.io/, but a quicker and more accessible way is to use an existing service platform.

[7]Was not yet open source when we were doing the benchmarking, and was later on also introduced in https://arxiv.org/abs/1805.10190.

(1) All four platforms support Intent classification and NER; (2) None of them support Multiple Intents where a single utterance might express more than one Intent, i.e. is performing more than one action. This is potentially a significant limitation because such utterances are generally very common in spoken dialogue; (3) Particular Entities and Entity types tend to be *dependent* on particular Intent types, e.g. with a 'set_alarm' intent one would expect a time stamp as its argument. Therefore we think that joint models, or models that treat Intent & Entity classification together would perform better. We were unable to ascertain this for any of the commercial systems, but Rasa treats them independently (as of Dec 2018). (4) None of the platforms use dialogue context for Intent classification and NER—this is another significant limitation, e.g. in understanding elliptical or fragment utterances which depend on the context for their interpretation.

4 Data Collection and Annotation

The evaluation of NLU services was performed in the context of building a SDS, aka Conversational Interface, for a home assistant robot. The home robot is expected to perform a wide variety of tasks, ranging from setting alarms, playing music, search, to movie recommendation, much like existing commercial systems such as Microsoft's Cortana, Apple's Siri, Google Home or Amazon Alexa. Therefore the NLU component in a SDS for such a robot has to understand and be able to respond to a very wide range of user requests and questions, spanning multiple domains, unlike a single domain SDS which only understands and responds to the user in a specific domain.

4.1 Data Collection: Crowdsourcing Setup

To build the NLU component we collected real user data via Amazon Mechanical Turk (AMT). We designed tasks where the Turker's goal was to answer questions about how people would interact with the home robot, in a wide range of scenarios designed in advance, namely: alarm, audio, audiobook, calendar, cooking, datetime, email, game, general, IoT, lists, music, news, podcasts, general Q&A, radio, recommendations, social, food takeaway, transport, and weather.

The questions put to Turkers were designed to capture the different requests within each given scenario. In the 'calendar' scenario, for example, these pre-designed intents were included: 'set_event', 'delete_event' and 'query_event'. An example question for intent 'set_event' is: "How would you ask your PDA to schedule a meeting with someone?" for which a user's answer example was "Schedule a chat with Adam on Thursday afternoon". The Turkers would then type in their answers to these questions and select possible entities from the pre-designed suggested entities list for each of their answers. The Turkers didn't always follow the instructions fully, e.g. for the specified ‘delete_event’ Intent, an answer was: “PDA what is my next event?”; which clearly belongs to ‘query_event’ Intent. We have manually corrected all such errors either during post-processing or the subsequent annotations.

The data is organized in CSV format which includes information like scenarios, intents, user answers, annotated user answers etc.(See Table 4 in Appendix). The split training set and test set were converted into different JSON formats for each platform according to the specific requirements of the each platform (see Table 1)

Our final annotated corpus contains **25716 utterances, annotated for 64 Intents and 54 Entity Types**.

4.2 Annotation and Inter-annotator Agreement

Since there was a predetermined set of Intents for which we collected data, there was no need for separate Intent annotations(some Intent corrections were needed). We therefore only annotated the data for Entity Tokens & Entity Types. Three students were recruited to do the annotations. To calculate inter-annotator agreement, each student annotated the same set of 300 randomly selected utterances. Each student then annotated a third of the whole dataset, namely, about 8 K utterances for annotation. We used Fleiss’s Kappa, suitable for multiple annotators. A match was defined as follows: if there was any overlap between the Entity Tokens (i.e. Partial Tokens Matching), and the annotated Entity Types matched exactly. We achieved moderate agreement ($\kappa = 0.69$) for this task.

5 Evaluation Experiments

In this section we describe our evaluation experiments, comparing the performance of the four systems outlined above.

5.1 Train and Test Sets

Since LUIS caps the size of the training set to 10K, we chose 190 instances of each of the 64 Intents *at random*. Some of the Intents had slightly fewer instances than 190. This resulted in a **sub-corpus of 11036 utterances** covering all the 64 Intents and 54 Entity Types. The Appendix provides more details: Table 5 shows the number of the sentences for each Intent. Table 6 lists the number of entity samples for each Entity Type. For the evaluation experiments we report below, we performed *10 fold* *cross-validation* with 90% of the subcorpus for training and 10% for testing in each fold.[8]

5.2 System Versions and Configurations

Our latest evaluation runs were completed by the end of March 2018. The service API used was V1.0 for Dialogflow, V2.0 for LUIS. Watson API requests require data as a version parameter which is automatically matched to the closest internal version, where we specified `2017/04/21`.[9] In our conversational system we run the open source Rasa as our main NLU component because it allows us to have more control over further developments and extensions. The evaluation done for Rasa was on Version `0.10.5`, and we used its spacy_sklearn pipeline which uses Conditional Random Fields for NER and sk-learn (scikit-learn) for Intent classifications. Rasa also provides other built-in components for the processing pipeline, e.g. MITIE, or latest tensorflow_embedding pipeline.

6 Results and Discussion

We performed 10-fold cross validation for each of the platforms and pairwise t-tests to compare the mean F-scores of every pair of platforms. The results in Table 2 show the micro-average[10] scores for Intent and Entity Type classification over 10-fold cross validation. Table 3 shows the micro-average F-scores of each platform after combining the results of Intents and Entity Types. Tables 7 and 8 in the Appendix show the detailed confusion matrices used to calculate the scores of Precision, Recall and F1 for Intents and Entities.

Performing significance tests on separate Intent and Entity scores in Table 2 revealed: For Intent, there is no significant difference between Dialogflow, LUIS and Rasa. Watson F1 score (0.882) is significantly higher than other platforms ($p < 0.05$, with large or very large effects sizes—Cohen's D). However, for Entities, Watson achieves significantly lower F1 scores ($p < 0.05$, with large or very large effects sizes—Cohen's D) due to its very low Precision. One explanation for this is the high number of Entity candidates produced in its predictions, leading to a high number

[8] We also note here that our dataset was inevitably unbalanced across the different Intents & Entities: e.g. some Intents had much fewer instances: `iot_wemo` had only 77 instances. But this would affect the performance of the four platforms equally, and thus does not confound the results presented below.

[9] At the time of producing the camera-ready version of this paper, we noticed the seemingly recent addition of a 'Contextual Entity' annotation tool to Watson, much like e.g. in Rasa. Wed like to stress that this paper does *not* include an evaluation of this feature in Watson NLU.

[10] Micro-average sums up the individual TP, FP, and FN of all Intent/Entity classes to compute the average metric.

Table 2 Overall scores for intent and entity

	Intent			Entity		
	Prec	Rec	F1	Prec	Rec	F1
Rasa	0.863	0.863	0.863	0.859	0.694	0.768
Dialogflow	0.870	0.859	0.864	0.782	0.709	0.743
LUIS	0.855	0.855	0.855	0.837	0.725	**0.777**
Watson	0.884	0.881	**0.882**	0.354	0.787	**0.488**

Table 3 Combined overall scores

	Prec	Rec	F1
Rasa	0.862	0.787	0.822
Dialogflow	0.832	0.791	0.811
LUIS	0.848	0.796	0.821
Watson	0.540	0.838	**0.657**

of False Positives.[11] It also shows that there are significant differences for Entity F1 score between Dialogflow, LUIS and Rasa. LUIS achieved the top F1 score (0.777) on Entities.

Table 3 shows that all NLU services have quite close F1 scores except for Watson which had significantly lower score ($p < 0.05$, with large or very large effects sizes—Cohen's D) due to its lower entity score as discussed above. The significance test shows no significant differences between Dialogflow, LUIS and Rasa.

The detailed data analysis results in the Appendix (see Tables 5 and 6) for fold-1[12] reveal that distributions of Intents and Entities are imbalanced in the datasets. Also, our data contains some noisy Entity annotations, often caused by ambiguities, which our simplified annotation scheme was not able to capture. For example, an utterance in the pattern "play xxx please" where xxx could be any entity from song_name, audiobook_name, radio_name, posdcasts_name or game_name, e.g. "play space invaders please" which could be annotated the entity as [song_name: space invaders] or [game_name: space invaders]. This type of Intent ambiguity that can only be resolved by more sophisticated approaches that incorporate domain knowledge and the dialogue context. Nevertheless, despite the noisiness of the data, we believe that it represents a real-world use case for NLU engines.

[11]Interestingly, Watson only requires a list of possible entities rather than entity annotation in utterances as other platforms do (See Table 1).

[12]Tables for other folds are omitted for space reason.

7 Conclusion

The contributions of this paper are two-fold: First, we present and release a large NLU dataset in the context of a real-world use case of a home robot, covering 21 domains with 64 Intents and 54 Entity Types. Secondly, we perform a comparative evaluation on this data of some of the most popular NLU services—namely the commercial platforms Dialogflow, LUIS, Watson and the open source Rasa.

The results show they all have similar functions/features and achieve similar performance in terms of combined F-scores. However, when dividing out results for Intent and Entity Type recognition, we find that Watson has significant higher F-scores for Intent, but significantly lower scores for Entity Type. This was due to its high number of false positives produced in its Entity predictions. As noted earlier, we have *not* here evaluated Watson's recent 'Contextual Entity' annotation tool.

In future work, we hope to continuously improve the data quality and observe its impact on NLU performance. However, we do believe that noisy data presents an interesting real-world use-case for testing current NLU services. We are also working on extending the data set with spoken user utterances, rather than typed input. This will allow us to investigate the impact of ASR errors on NLU performance.

Appendix

We provide some examples of the data annotation and the training inputs to each of the 4 platforms in Table 4, Listings 1, 2, 3 and 4.

We also provide more details on the train and test data distribution, as well as the Confusion Matrix for the first fold (Fold_1) of the 10-Fold Cross Validation. Table 5 shows the number of the sentences for each Intent in each dataset. Table 6 lists the number of entity samples for each Entity Type in each dataset. Tables 7 and 8 show the confusion matrices used to calculate the scores of Precision, Recall and F1 for Intents and Entities. The TP, FP, FN and TN in the tables are short for True Positive, False Positive, False Negative and True Negative respectively.

Listing 1 Rasa train data example snippet

```
{
  "rasa_nlu_data": {
    "common_examples": [  {
        "text": "lower the lights in the
            bedroom",
        "intent": "iot_hue_lightdim",
        "entities": [   {
            "start": 24,
            "end": 31,
            "value": "bedroom",
            "entity": "house_place"
```

```
            } ]   },
        {
          "text": "dim the lights in my bedroom",
          "intent": "iot_hue_lightdim",
          "entities": [  {
              "start": 21,
              "end": 28,
              "value": "bedroom",
              "entity": "house_place"
            } ]    },
         ... ...
        ]
}
```

Table 4 Data annotation example snippet

Userid	Answerid	Scenario	Intent	Answer_annotation
1	2	Alarm	Set	Wake me up at [time: nine am] on [date: friday]
2	558	Alarm	Remove	Cancel my [time: seven am] alarm
2	559	Alarm	Remove	Remove the alarm set for [time: ten pm]
2	561	Alarm	Query	What alarms i have set
502	12925	Calendar	Query	What is the time for [event_name: jimmy's party]
653	17462	Calendar	Query	What is up in my schedule [date: today]
2	564	Calendar	Remove	Please cancel all my events for [date: today]
2	586	Play	Music	I'd like to hear [artist_name: queen's] [song_name: barcelona]
65	2813	Play	Radio	Play a [radio_name: pop station] on the radio
740	19087	Play	Podcasts	Play my favorite podcast
1	1964	Weather	Query	Tell me the weather in [place_name: barcelona] in [time: two days from now]
92	3483	Weather	Query	What is the current [weather_descriptor: temperature] outside
394	10448	Email	Sendemail	Send an email to [person: sarah] about [event_name: brunch] [date: today]
4	649	Email	Query	Has the [business_name: university of greenwich] emailed me
2	624	Takeaway	Order	Please order some [food_type: sushi] for [meal_type: dinner]
38	2045	Takeaway	Query	Search if the [business_type: restaurant] does [order_type: take out]

Table 5 Data distribution for intents in Fold_1

Intent	Total	Train	Test	Intent	Total	Train	Test	Intent	Total	Train	Test
Alarm_query	194	175	19	General_negate	194	175	19	Play_music	194	175	1
Alarm_remove	117	106	11	General_praise	194	175	19	Play_podcasts	194	175	1
Alarm_set	194	175	19	General_quirky	194	175	19	Play_radio	194	175	19
Audio_volume_down	80	72	8	General_repeat	194	175	19	qa_currency	194	175	19
Audio_volume_mute	157	142	15	iot_cleaning	167	151	16	qa_definition	194	175	19
Audio_volume_up	139	126	13	iot_coffee	194	175	19	qa_factoid	194	175	19
Calendar_query	194	175	19	iot_hue_lightchange	194	175	19	qa_maths	148	134	14
Calendar_remove	194	175	19	iot_hue_lightdim	126	114	12	qa_stock	194	175	19
Calendar_set	194	175	19	iot_hue_lightoff	194	175	19	rec_events	194	175	19
Cooking_recipe	194	175	19	iot_hue_lighton	38	35	3	rec_locations	194	175	19
Datetime_convert	87	79	8	iot_hue_lightup	140	126	14	rec_movies	107	97	10
Datetime_query	194	175	19	iot_wemo_off	98	89	9	Social_post	194	175	19
Email_addcontact	87	79	8	iot_wemo_on	76	69	7	Social_query	183	165	18
Email_query	194	175	19	Lists_createoradd	194	175	19	Takeaway_order	194	175	19
Email_querycontact	194	175	19	Lists_query	194	175	19	Takeaway_query	194	175	19
Email_sendemail	194	175	19	Lists_remove	194	175	19	Transport_query	194	175	19
General_affirm	194	175	19	Music_likeness	180	162	18	Transport_taxi	181	163	18
General_commandstop	194	175	19	Music_query	194	175	19	Transport_ticket	194	175	19
General_confirm	194	175	19	Music_settings	77	70	7	Transport_traffic	190	171	19
General_dontcare	194	175	19	News_query	194	175	19	Weather_query	194	175	19
General_explain	194	175	19	Play_audiobook	194	175	19				
General_joke	122	110	12	Play_game	194	175	19				

Table 6 Data distribution for entities in Fold_1

Entity	Trainset	Testset	Entity	Trainset	Testset	Entity	Trainset	Testset
Alarm_type	14	0	Event_name	352	48	Person	468	42
App_name	32	5	Food_type	302	25	Personal_info	100	14
Artist_name	91	11	Game_name	133	17	Place_name	869	95
Audiobook_author	10	1	Game_type	1	0	Player_setting	190	19
Audiobook_name	97	10	General_frequency	27	5	Playlist_name	22	1
Business_name	394	41	House_place	259	25	Podcast_descriptor	67	6
Business_type	199	19	Ingredient	17	4	Podcast_name	44	2
Change_amount	57	9	Joke_type	59	4	Radio_name	99	12
Coffee_type	31	4	List_name	211	13	Relation	127	13
Color_type	135	11	Meal_type	37	0	Song_name	51	9
Cooking_type	10	0	Media_type	370	40	Time	511	62
Currency_name	296	35	Movie_name	18	0	Time_zone	59	7
Date	905	85	Movie_type	13	0	Timeofday	150	26
Definition_word	158	16	Music_album	1	0	Transport_agency	59	10
Device_type	353	41	Music_descriptor	17	2	Transport_descriptor	11	0
Drink_type	6	0	Music_genre	72	8	Transport_name	10	2
Email_address	38	5	News_topic	75	9	Transport_type	363	35
Email_folder	17	1	Order_type	151	17	Weather_descriptor	95	14

Table 7 Confusion matrix summary for intents in Fold_1

	Rasa				Dialogflow				LUIS				Watson			
Intent	TP	FP	FN	TN	TP	FP	FN	TN	TP	FP	FN	TN	TP	FP	FN	TN
Alarm_query	17	1	2	1056	19	0	0	1057	18	2	1	1055	19	0	0	1057
Alarm_remove	11	0	0	1065	10	2	1	1063	9	0	2	1065	11	0	0	1065
Alarm_set	18	3	1	1054	17	4	2	1053	17	3	2	1054	17	3	2	1054
Audio_volume_down	7	1	1	1067	8	0	0	1068	7	0	1	1068	8	0	0	1068
Audio_volume_mute	13	1	2	1060	14	0	1	1061	12	1	3	1060	14	1	1	1060
Audio_volume_up	12	3	1	1060	13	0	0	1063	12	3	1	1060	12	3	1	1060
Calendar_query	11	10	8	1047	13	18	6	1039	11	6	8	1051	10	8	9	1049
Calendar_remove	17	0	2	1057	18	1	1	1056	18	2	1	1055	19	1	0	1056
Calendar_set	16	2	3	1055	14	2	5	1055	14	4	5	1053	16	3	3	1054
Cooking_recipe	15	1	4	1056	11	2	8	1055	13	4	6	1053	15	1	4	1056
Datetime_convert	5	2	3	1066	7	4	1	1064	7	2	1	1066	8	2	0	1066
Datetime_query	17	4	2	1053	18	9	1	1048	17	4	2	1053	18	4	1	1053
Email_addcontact	8	3	0	1065	8	0	0	1068	8	0	0	1068	8	2	0	1066
Email_query	17	1	2	1056	18	1	1	1056	15	3	4	1054	17	2	2	1055
Email_querycontact	11	4	8	1053	13	3	6	1054	14	4	5	1053	14	3	5	1054
Email_sendemail	17	1	2	1056	16	1	3	1056	16	4	3	1053	17	2	2	1055

(continued)

Table 7 (continued)

	Rasa				Dialogflow				LUIS				Watson			
General_affirm	19	1	0	1056	19	0	0	1057	19	0	0	1057	19	1	0	1056
General_commandstop	19	0	0	1057	18	1	1	1056	19	0	0	1057	19	1	0	1056
General_confirm	19	1	0	1056	19	0	0	1057	19	0	0	1057	19	0	0	1057
General_dontcare	19	0	0	1057	19	1	0	1056	18	1	1	1056	19	2	0	1055
General_explain	19	1	0	1056	19	0	0	1057	18	0	1	1057	19	2	0	1055
General_joke	11	0	1	1064	12	0	0	1064	12	0	0	1064	12	0	0	1064
General_negate	18	0	1	1057	19	0	0	1057	19	1	0	1056	19	0	0	1057
General_praise	18	1	1	1056	19	0	0	1057	19	1	0	1056	18	1	1	1056
General_quirky	11	22	8	1035	4	2	15	1055	8	16	11	1041	7	9	12	1048
General_repeat	19	0	0	1057	19	1	0	1056	19	0	0	1057	19	0	0	1057
iot_cleaning	14	1	2	1059	13	6	3	1054	16	1	0	1059	16	1	0	1059
iot_coffee	18	3	1	1054	18	1	1	1056	18	0	1	1057	19	1	0	1056
iot_hue_lightchange	15	1	4	1056	14	3	5	1054	15	4	4	1053	13	3	6	1054
iot_hue_lightdim	12	0	0	1064	11	0	1	1064	10	1	2	1063	11	1	1	1063
iot_hue_lightoff	17	2	2	1055	15	0	4	1057	17	1	2	1056	17	2	2	1055
iot_hue_lighton	3	3	0	1070	3	3	0	1070	2	3	1	1070	3	3	0	1070
iot_hue_lightup	9	1	5	1061	11	1	3	1061	11	0	3	1062	11	2	3	1060
iot_wemo_off	9	2	0	1065	8	4	1	1063	9	4	0	1063	9	2	0	1065
iot_wemo_on	5	2	2	1067	5	1	2	1068	4	3	3	1066	6	1	1	1068
Lists_createoradd	16	2	3	1055	16	6	3	1051	16	5	3	1052	18	3	1	1054
Lists_query	16	3	3	1054	16	5	3	1052	16	3	3	1054	14	2	5	1055
Lists_remove	17	1	2	1056	18	3	1	1054	18	2	1	1055	18	0	1	1057

(continued)

Table 7 (continued)

	Rasa				Dialogflow				LUIS				Watson			
Music_likeness	12	4	6	1054	13	5	5	1053	13	3	5	1055	14	1	4	1057
Music_query	13	0	6	1057	11	3	8	1054	10	4	9	1053	11	2	8	1055
Music_settings	6	2	1	1067	4	2	3	1067	7	0	0	1069	7	2	0	1067
News_query	13	9	6	1048	10	4	9	1053	13	3	6	1054	14	1	5	1056
Play_audiobook	16	3	3	1054	13	8	6	1049	17	1	2	1056	16	2	3	1055
Play_game	15	5	4	1052	13	2	6	1055	13	2	6	1055	13	2	6	1055
Play_music	13	4	6	1053	16	5	3	1052	12	11	7	1046	12	14	7	1043
Play_podcasts	17	0	2	1057	14	1	5	1056	16	0	3	1057	17	1	2	1056
Play_radio	15	1	4	1056	15	2	4	1055	17	1	2	1056	15	2	4	1055
qa_currency	17	1	2	1056	16	0	3	1057	18	0	1	1057	18	0	1	1057
qa_definition	19	0	0	1057	13	2	6	1055	18	0	1	1057	18	1	1	1056
qa_factoid	10	13	9	1044	7	9	12	1048	15	15	4	1042	14	8	5	1049
qa_maths	14	2	0	1060	12	2	2	1060	13	4	1	1058	14	1	0	1061
qa_stock	19	2	0	1055	19	1	0	1056	19	0	0	1057	19	1	0	1056
Recommendation_events	13	2	6	1055	14	6	5	1051	16	3	3	1054	15	2	4	1055
Recommendation_locations	16	1	3	1056	15	1	4	1056	17	2	2	1055	16	1	3	1056
Recommendation_movies	8	2	2	1064	8	2	2	1064	9	1	1	1065	10	2	0	1064
Social_post	18	3	1	1054	17	4	2	1053	18	1	1	1056	19	1	0	1056
Social_query	16	5	2	1053	14	8	4	1050	17	3	1	1055	17	3	1	1055
Takeaway_order	12	0	7	1057	16	2	3	1055	16	4	3	1053	16	1	3	1056
Takeaway_query	18	6	1	1051	19	3	0	1054	16	2	3	1055	18	3	1	1054
Transport_query	16	3	3	1054	17	3	2	1054	13	3	6	1054	14	5	5	1052
Transport_taxi	17	2	1	1056	17	1	1	1057	18	0	0	1058	18	1	0	1057
Transport_ticket	16	1	3	1056	17	0	2	1057	16	1	3	1056	16	2	3	1055
Transport_traffic	18	1	1	1056	18	1	1	1056	18	1	1	1056	19	2	0	1055
Weather_query	16	2	3	1055	12	2	7	1055	13	5	6	1052	13	2	6	1055

Table 8 Confusion matrix summary for entities in Fold_1

	Rasa				Dialogflow				LUIS				Watson			
Entity	TP	FP	FN	TN	TP	FP	FN	TN	TP	FP	FN	TN	TP	FP	FN	TN
App_name	3	0	2	1071	2	1	3	1070	3	0	2	1071	4	10	1	1061
Artist_name	3	0	8	1065	5	1	6	1064	4	2	7	1063	3	1	8	1064
Audiobook_author	0	0	1	1075	0	0	1	1075	0	0	1	1075	0	0	1	1075
Audiobook_name	2	3	8	1063	6	2	4	1064	5	1	5	1065	6	3	4	1063
Business_name	25	12	16	1027	32	8	9	1029	32	5	9	1031	29	30	12	1008
Business_type	15	2	4	1055	13	1	6	1056	14	5	5	1054	16	45	3	1014
Change_amount	7	0	2	1067	6	2	3	1065	8	2	1	1065	6	12	3	1056
Coffee_type	1	0	3	1072	2	1	2	1071	2	0	2	1072	2	4	2	1068
Color_type	8	2	3	1063	8	1	3	1064	8	1	3	1064	9	26	2	1042
Currency_name	25	0	10	1058	14	0	21	1058	28	4	7	1056	31	12	4	1049
Date	77	8	8	983	74	25	11	969	78	9	7	984	80	30	5	971
Definition_word	7	2	9	1058	10	3	6	1057	11	4	5	1056	6	104	10	961
Device_type	33	0	8	1035	24	10	17	1027	33	6	8	1029	38	76	3	963
Email_address	4	0	1	1071	4	1	1	1070	3	2	2	1071	1	0	4	1071
Email_folder	1	0	0	1075	1	0	0	1075	1	0	0	1075	1	0	0	1075
Event_name	27	4	21	1024	25	25	23	1005	24	6	24	1023	30	56	18	973
Food_type	13	3	12	1048	16	5	9	1046	16	4	9	1047	17	16	8	1040
Game_name	7	2	10	1057	11	2	6	1057	12	0	5	1059	9	2	8	1057
General_frequency	1	1	4	1070	0	0	5	1071	2	0	3	1071	3	3	2	1069
House_place	22	1	3	1050	22	10	3	1042	24	1	1	1050	25	18	0	1033
Ingredient	0	0	4	1072	1	0	3	1072	0	1	4	1072	1	3	3	1069

(continued)

Table 8 (continued)

	Rasa				Dialogflow				LUIS				Watson			
Joke_type	3	1	1	1071	3	0	1	1072	3	2	1	1070	2	53	2	1019
List_name	9	7	4	1056	6	2	7	1061	10	5	3	1058	7	56	6	1010
Media_type	29	4	11	1033	26	24	14	1013	31	11	9	1026	34	81	6	961
Music_descriptor	0	0	2	1074	0	0	2	1074	0	0	2	1074	0	4	2	1070
Music_genre	6	1	2	1067	7	2	1	1066	6	1	2	1067	7	8	1	1060
News_topic	0	2	9	1065	3	3	6	1064	2	4	7	1063	3	18	6	1049
Order_type	14	3	3	1056	12	3	5	1056	13	2	4	1057	17	8	0	1051
Person	31	14	11	1021	31	12	11	1023	30	7	12	1028	27	36	15	999
Personal_info	5	0	9	1063	5	1	9	1062	7	4	7	1059	12	58	2	1011
Place_name	65	22	30	971	66	17	29	976	71	5	24	986	76	39	19	961
Player_setting	13	2	6	1056	9	3	10	1055	16	7	3	1052	18	71	1	988
Playlist_name	0	0	1	1075	0	0	1	1075	0	0	1	1075	0	0	1	1075
Podcast_descriptor	5	1	1	1069	4	1	2	1069	5	2	1	1068	5	9	1	1061
Podcast_name	0	0	2	1074	0	0	2	1074	1	2	1	1072	0	111	2	968
Radio_name	4	2	8	1063	6	2	6	1062	7	5	5	1060	2	17	10	1048
Relation	8	0	5	1063	6	4	7	1059	7	1	6	1063	10	4	3	1059
Song_name	4	1	5	1066	5	2	4	1065	3	1	6	1066	3	13	6	1055
Time	53	3	9	1013	45	18	17	1002	49	12	13	1010	55	119	7	928
Time_zone	2	0	5	1071	3	1	4	1070	2	1	5	1070	6	63	1	1019
Timeofday	23	3	3	1047	13	3	13	1047	22	4	4	1047	26	4	0	1046
Transport_agency	10	0	0	1066	10	0	0	1066	10	0	0	1066	10	0	0	1066
Transport_name	0	0	2	1074	0	0	2	1074	0	0	2	1074	0	0	2	1074
Transport_type	35	1	0	1040	14	1	21	1041	34	4	1	1039	35	7	0	1035
Weather_descriptor	5	1	9	1063	7	3	7	1061	7	2	7	1062	8	12	6	1053

Listing 2 LUIS train data example snippet

```
{
  "intents": [
    {  "name": "play_podcasts"   },
    {  "name": "music_query"     },
    .......
  ],
  "entities": [  {
      "name": "Hier2",
      "children": [
        "business_type",  "event_name",  "place_name",  "time", "timeofday" ] },
    ... ...
  ],
  "utterances": [  {
      "text": "call a taxi for me",
      "intent": "transport_taxi",
      "entities": [ {
          "startPos": 7,
          "endPos": 10,
          "value": "taxi",
          "entity": "Hier9::transport_type"
        }     ]    },
     ... ...
    ]
}
```

Listing 3 Watson train data example snippet

```
---- Watson Entity list ----

joke_type,nice  joke_type,funny joke_type,sarcastic
 ... ...
relation,mum    relation,dad    person,ted
... ...
person,emma     person,bina     person,daniel bell

---- Watson utterance and Intent list ----

give me the weather for merced at three pm,weather_query
weather this week,weather_query
```

```
find weather report,weather_query
should i wear a hat today,weather_query
what should i wear is it cold outside,weather_query
is it going to snow tonight,weather_query
```

Listing 4 Dialogflow train data example snippet

```
---- Dialogflow Entity list ----
{
  "id": "... ...",
  "name": "artist_name",
  "isOverridable": true,
  "entries": [   {
      "value": "aaron carter",
      "synonyms": [
        "aaron carter"
      ] },
    {
      "value": "adele",
      "synonyms": [ "adele"    ]
    }  ],
  "isEnum": false,
  "automatedExpansion": true
}

---- Dialogflow "alarm_query" Intent annotation ----
{
  "userSays": [ {
      "id": " ... ... ",
      "data": [ {  "text": "checkout "   },
        {
          "text": "today",
          "alias": "date",
          "meta": "@date",
          "userDefined": true
        },
        {  "text": " alarm of meeting"    }
      ],
      "isTemplate": false,
      "count": 0
    },
   ... ...
] }
```

References

1. Braun D, Mendez AH, Matthes F, Langen M (2017) Evaluating natural language understanding services for conversational question answering systems. In: Proceedings of SIGDIAL 2017, pp 174–185
2. Canonico M, Russis LD (2018) A comparison and critique of natural language understanding tools. In: Proceedings of CLOUD COMPUTING 2018
3. Coucke A, Ball A, Delpuech C, Doumouro C, Raybaud S, Gisselbrecht T, Dureau J (2017) benchmarking natural language understanding systems: Google, Facebook, Microsoft, Amazon, and Snips. https://medium.com/snips-ai/benchmarking-natural-language-understanding-systems-google-facebook-microsoft-and-snips-2b8ddcf9fb19
4. Canh NT (2018) Benchmarking intent classification services. June 2018. https://medium.com/botfuel/benchmarking-intent-classification-services-june-2018-eb8684a1e55f
5. Wisniewski C, Delpuech C, Leroy D, Pivan F, Dureau J (2017) Benchmarking natural language understanding systems. https://snips.ai/content/sdk-benchmark-visualisation/

Dialogue System Live Competition: Identifying Problems with Dialogue Systems Through Live Event

Ryuichiro Higashinaka, Kotaro Funakoshi, Michimasa Inaba, Yuiko Tsunomori, Tetsuro Takahashi, and Reina Akama

Abstract We organized a competition entitled "the dialogue system live competition" in which the audience, consisting mainly of researchers in the dialogue community, watched and evaluated a live dialogue conducted between users and dialogue systems. The motivation behind the event was to cultivate state-of-the-art techniques in dialogue systems and enable the dialogue community to share the problems with current dialogue systems. There are two parts to the competition: preliminary selection and live event. In the preliminary selection, eleven systems were evaluated by crowd-sourcing. Three systems proceeded to the live event to perform dialogues with designated speakers and to be evaluated by the audience. This paper describes the design and procedure of the competition, the results of the preliminary selection and live event of the competition, and the problems we identified from the event.

R. Higashinaka (✉)
NTT Media Intelligence Laboratories, NTT Corporation, Yokosuka, Japan
e-mail: ryuichiro.higashinaka.tp@hco.ntt.co.jp

K. Funakoshi
Laboratory for Future Interdisciplinary Research of Science and Technology, Tokyo Institute of Technology, Yokohama, Japan
e-mail: funakoshi@lr.pi.titech.ac.jp

M. Inaba
Artificial Intelligence eXploration Research Center, The University of Electro-Communications, Chofu, Japan
e-mail: m-inaba@uec.ac.jp

Y. Tsunomori
Service Innovation Department, NTT DOCOMO INC., Tokyo, Japan
e-mail: yuiko.tsunomori.fc@nttdocomo.com

T. Takahashi
AI Deployment Project, Fujitsu Laboratories, LTD., Kawasaki, Japan
e-mail: takahashi.tet@jp.fujitsu.com

R. Akama
Graduate School of Information Sciences, Tohoku University, Sendai, Japan
e-mail: reina.a@ecei.tohoku.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_16

1 Introduction

As in many fields in computer science (e.g., question answering, document retrieval, and speech recognition), evaluation workshops have been a driving force in dialogue systems. There are two types of evaluation workshops; one for developing a specific module in a dialogue system and the other for improving the quality of an entire dialogue system.

In the former, organizers prepare datasets and participants develop their own algorithms, competing for the best performance with regards to the dataset. The Dialog State Tracking Challenge (DSTC) [9] has been a popular venue for task-oriented dialogue systems. In the DSTC, participants develop algorithms for spoken language understanding from dialogue logs. The Dialogue Breakdown Detection Challenges (DBDCs) [4] and NTCIR short text conversation [6] have been held in recent years for non-task-oriented dialogue systems. In the latter, participants develop dialogue systems and organizers prepare human evaluators for subjective evaluation. Popular challenges are the Spoken Dialogue Challenge [2], Loebner Prize (Turing test), Alexa Prize,[1] and the Conversational Intelligence Challenge.[2]

In this paper, we describe the event "the dialogue system live competition" we organized in Japan, which falls into the latter category. However, this event differs from previous ones as follows.

- Dialogues between human users and dialogue systems developed by participants are performed live in front of an audience, including many researchers and engineers, who watch the dialogues and evaluate the dialogue systems.
- After the live dialogues, the developers describe their systems and conduct a question-answering (QA) session with the audience.
- Before and after the event, questionnaires are handed out, asking for what the audience thinks the current problems are with current dialogue systems.

By having the audience watch and evaluate a dialogue live, we expect that the dialogue community can share the same understanding about the problems with current dialogue systems, which we believe is important for the community. We give details on the design of the competition, results of the preliminary selection and live event, and problems we identified with dialogue systems from the live event.

2 Design of Live Competition

The aim of the live competition is to have the dialogue community share the same understanding about the current problems with dialogue systems. To this end, we need to have systems that exhibit state-of-the-art or at least high-quality performance to be used in the live event; we need to ask developers to submit their dialogue systems

[1] https://developer.amazon.com/alexaprize.

[2] http://convai.io/.

and conduct a preliminary selection. We also need to define evaluation metrics and regulations. This section describes the design (specifications) of the competition.

2.1 Target Dialogue Systems

The competition focuses on chat-oriented dialogue systems. One reason for avoiding targeting task-oriented dialogue systems is that chat-oriented ones are still in the phase of basic research, making it easy for both academia and industry to participate and submit their systems. Since task-oriented dialogue systems benefit from the chat function from a social perspective, we also expected the submission of systems from those working on task-oriented dialogue systems. Another reason for avoiding targeting task-oriented dialogue systems is that, if we focus on a certain task, only the people working on that task may be able to participate, limiting the number of participants. This is especially true when considering the high cost of constructing an entire dialogue system.

2.2 Platform

The dialogue systems for the competition are implemented as bots on the Telegram platform. Telegram is an online messaging system developed by Telegram Messenger LLP and has been used as a platform for the Conversational Intelligence Challenge. To have a conversation with a chatbot, users just need to specify the `username` of the chatbot and start the conversation.

Participating teams of the competition create their chat-oriented dialogue systems and make the systems accessible on Telegram. Systems are submitted by informing the organizers of the username of the systems. Although Telegram is equipped with various APIs (in Python, Java, Ruby, etc.) with which to support the development of chatbots, we also created several examples and how-to web pages to facilitate system development.

2.3 Requirements

We set the following requirements for systems to be submitted to the competition.

1. The language of the system is Japanese.
2. When the system receives a `/start` message on the Telegram platform, the system initiates a dialogue.
3. Turns alternate between the system and user, and the system can produce just one utterance (one speech bubble) in its turn.

4. The content of the utterance can only include text and should not include emoticons, facial expressions, and stamps.
5. The utterance should not include line breaks.
6. The system should not use profile pictures, icons, and profile descriptions.
7. Upon receiving the 15th user utterance, the system needs to end the dialogue by its 16th utterance and show a unique dialogue ID, which is used by organizers for evaluation.

We devised requirement 3 to conduct the evaluation independent of the interaction style of users. We devised requirements 4–6 so that the evaluation can be conducted on the basis of the textual content of the dialogue. Note that, in the preliminary selection using crowd-workers, we also ask the workers to satisfy the same requirements when making their utterances. We set the number of system utterances to 16, which is reasonably long when considering the ratio of dialogue breakdowns [3] in current chat-oriented dialogue systems.

2.4 Evaluation Metric

Although chat has many functions and it is difficult to decide on an appropriate evaluation metric, we use a single metric "how much the user is willing to chat with the system again" in the competition because one of the most important functions of chat-oriented dialogue systems is to maintain the relationship with users. The choice of this criterion is inspired by Alexa Prize, which uses basically the same metric. The evaluation was done on a five-point Likert scale. Note that, the users are explicitly notified that they are talking to a system (not a human) before evaluation.

2.5 Preliminary Selection and Live Event

The preliminary selection and live event are conducted as follows:

Preliminary selection We use crowd-workers in the preliminary selection. We ask them to sign in to Telegram and chat with their assigned chatbots by using the evaluation metric described above. Since the evaluation of a chat-oriented dialogue system can be very subjective, we use multiple workers (we used at most 30 workers; see the next section) for each system. The averaged score by the workers is used for ranking the systems.

Live event The systems that performed well in the preliminary selection can proceed to the live event. In the live event, the designated speakers talk to the systems, and the audience watches and evaluates the systems. After that, the developers present their systems and answer questions from the audience. On the basis of the average scores of the audience, system ranking is determined.

2.6 Information Disclosure

To encourage participation, especially from the industry, we set a policy for information disclosure. For the teams who have passed the preliminary selection, we disclose their names/institutions. For others, the names are anonymized in principle; we will only disclose them if the developers opt in to be disclosed. The dialogue data collected in the preliminary selection will not be disclosed in principle, although those willing to make the data public can do so by contacting the organizers. The dialogue data in the live event will be made public on the competition's website.[3]

3 Preliminary Selection

We now describe the preliminary selection together with the results. We also present analyses of the results and describe the methods that may be promising on the basis of the results.

3.1 Entries

The live competition was announced in June 2018 and entries were closed at the end of September 2018. We had 12 entries (from 12 teams), including one from the organizers. The institutions of the participating teams included those from both academia and industry. Refer to Table 1 for the participating teams.

The system that we (the organizers) prepared is called IRS, which is based on the IR-STATUS approach described by Ritter et al. [5]. This is a retrieval-based approach: it first searches for an utterance that is most similar to the input user utterance in the database (we use 26,972 utterance pairs obtained from human-human chat dialogue data). Then, the response associated with that utterance is used as an output system utterance. Lucene[4] with the default setting is used for retrieval. For morphological analysis, we use JapaneseAnalyzer, which comes with Lucene.

3.2 Evaluation Through Crowd-Sourcing

The evaluation process started on October 1st. Since one of the systems could not be tested by the organizers, we conducted the evaluation on the remaining 11 systems. We used CrowdWorks[5] as the crowd-sourcing platform. We recruited crowd-workers

[3] https://dialog-system-live-competition.github.io/dslc1/.

[4] http://lucene.apache.org/.

[5] https://crowdworks.jp/.

who had good records in their previous tasks; namely, those with over 95% task-acceptance rate could take part in the evaluation. 300 Japanese yen (two to three US dollars) was paid on the completion of the task.

Since the systems operate on the Telegram platform, we created a crowd-sourcing task for each system. Each crowd-worker was only able to talk to a system once, although it was possible for them to talk to different systems. To avoid relative evaluation, we asked the crowd-workers to rate the systems by absolute evaluation and not to compare among systems.

After conducting a dialogue, the crowd-workers rated the system by showing their degree of agreement with the following statement: "I'm willing to chat with the system again" on a five-point Likert scale; 1 indicates "strongly agree" and 5 indicates "strongly disagree". We also solicited free-style comments from the crowd-workers about dialogue and system quality. The crowd-workers completed the task by sending us the dialogue logs, evaluation scores, and comments.

We assigned 30 crowd-workers for each system. Since some workers did not follow our guideline, some dialogues were regarded as invalid. As a result, each dialogue system was rated by between 26 to 30 crowd-workers.

3.3 Results

Table 1 shows the scores averaged over the raters. The table also shows the institutions, team names, and Telegram bot names. The anonymized participants are called TEAM1–6. Since there was a notable difference between the third and fourth places, we decided that the top three teams would proceed to the live event. The box plot of the scores for each team is shown in Fig. 1. We also conducted a non-parametric multiple comparison test (Steel-Dwass test) and found that only the top three teams, who proceeded to the live event, statistically outperformed RSL. In addition, only NTTdocomo showed marginal statistical significance over TEAM3.

3.4 Analysis

Table 2 summarizes the systems. The top-ranked teams use the combination of rule- and generation-based approaches. The mid-ranked teams use rule- and retrieval-based approaches. The low-ranked teams use only the rule- or generation-based approach. Therefore, using rules is important but needs to be complemented with the generation-based approach. It is also clear that the top-ranked teams use an abundance of knowledge. It is surprising that neural-based methods struggle; it seems that there are still limitations to appropriately using neural-based methods within a dialogue.

Table 1 Results of preliminary selection: averaged scores from crowd-workers

Rank	Score	Institution	Team	Telegram bot name
1	1.97	NTT DOCOMO INC.	NTTdocomo	MarikoZatsudanBot
2	2.10	NTT Communication Science Labs.	NTTCS	tripfreak
3	2.27	Tohoku University	teamzunko	zunko
4	2.57	Organizers	IRS	IRS
5	2.72	Anonymous	TEAM1	Anonymous
6	2.73	Anonymous	TEAM2	Anonymous
8	2.75	Anonymous	TEAM3	Anonymous
7	2.76	Anonymous	TEAM4	Anonymous
9	3.31	Waseda University	RSL	momokoBot
10	4.10	Anonymous	TEAM5	Anonymous
11	4.77	Anonymous	TEAM6	Anonymous

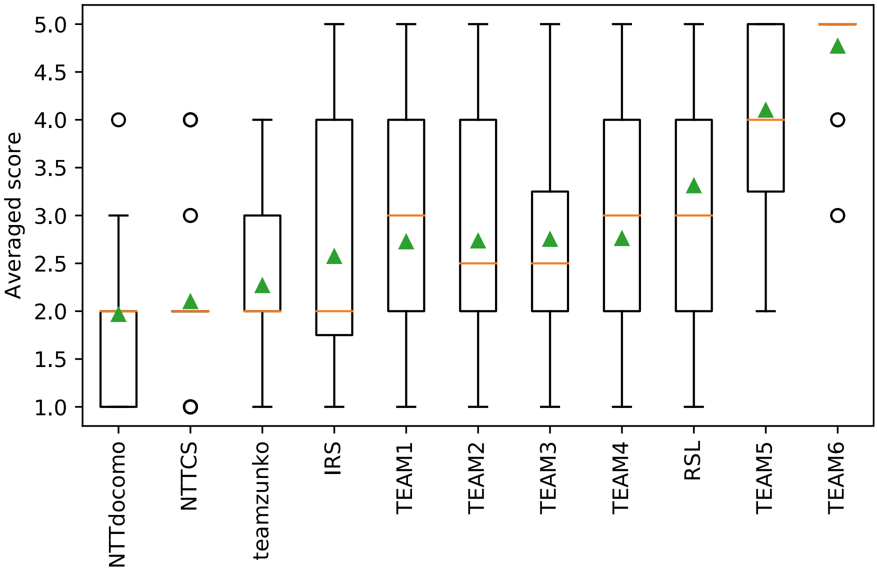


Fig. 1 Box plot of scores for each participating team

Table 3 shows the statistics regarding user and system utterances. To calculate the number of words, we used MeCab[6] for morphological analysis. Although the length

[6]http://taku910.github.io/mecab/.

Table 2 Description of each system as reported by participating teams. Approach can be rule-, retrieval-, or generation-based. Knowledge used can be large text data, knowledge base (KB), or dialogue data. N/A was used when no machine-learning (ML) was used

Rank	Team	Approach			Knowledge			
		Rule	Retrieval	Generation	Text	KB	Dialogue	ML algorithm
1	NTTdocomo	✓		✓	✓			Other
2	NTTCS	✓			✓	✓	✓	CRF, SVM
3	teamzunko	✓		✓	✓	✓	✓	RNN
4	IRS		✓				✓	N/A
5	TEAM1	✓	✓				✓	N/A
6	TEAM2	✓		✓	✓			CNN, RNN
7	TEAM3	✓	✓		✓		✓	CNN
8	TEAM4	✓	✓				✓	RNN
9	RSL	✓						N/A
10	TEAM5			✓			✓	RNN
11	TEAM6			✓			✓	Transformer

Table 3 Statistics regarding system and user utterances. Chars/words indicate averaged number of characters/words per utterance. Uniq indicates ratio of unique words

Rank	Team	No. dialogues	System utterance			User utterance		
			Chars	Words	Uniq (%)	Chars	Words	Uniq (%)
1	NTTdocomo	29	22.99	13.46	18.96	15.51	9.40	23.86
2	NTTCS	30	45.34	27.16	3.39	15.84	9.41	16.81
3	teamzunko	30	40.78	25.05	10.43	13.09	7.94	21.25
4	IRS	28	30.30	18.08	19.09	16.97	10.06	22.56
5	TEAM1	29	32.27	18.65	14.89	13.59	8.05	21.77
6	TEAM2	30	17.62	10.09	6.98	14.78	8.70	19.76
7	TEAM3	28	15.23	8.03	21.46	15.34	8.97	23.18
8	TEAM4	29	13.96	8.25	21.07	16.89	10.13	20.63
9	RSL	29	57.44	30.70	3.42	11.59	6.31	18.71
10	TEAM5	30	9.82	6.23	1.46	11.50	6.68	20.81
11	TEAM6	26	4.36	1.93	1.24	11.59	7.12	1.94

of system utterances differs from system to system, longer utterances are generally preferred by users. We also found that the ratio of unique words needs not be high (see Uniq of NTTCS); it seems satisfactory dialogue can be achieved by fixed pattern-like utterances.

4 Live Event

The teams that proceeded to the live event on 21st November 2018 were NTTdocomo, NTTCS, and teamzunko. The system descriptions are as follows; readers capable of reading Japanese are referred to individual system papers [1, 7, 8]. See also Fig. 2 for example dialogues.

NTTdocomo Their open-domain chat-oriented dialogue system estimates the focus and dialogue act from the user utterance and produces a system utterance on the basis of a large utterance database and rules.

NTTCS Their chat-oriented dialogue system talks about travel. On the basis of the manually constructed dialogue flow and the utterance knowledge created semi-automatically, it understands/generates appropriate utterances depending on the context.

teamzunko Their system uses three types of knowledge (Wikipedia-based knowledge base, quiz QA dataset, and puns scraped from the web) to generate rule-based

NTTdocomo	S: Hi, my name is Mariko. How have you been? U: I've been preparing for the workshop. S: Now, cosmetics manufacturer are keen on anti-aging products. U: I don't know much about cosmetics. S: Lately, males are also keen on cosmetics. U: Oh, I didn't know that. S: I guess so.
NTTCS	S: Hello, could you tell me your name? U: My name is Den. S: Your name is Den. I'm Riko. Nice to meet you. Where do you live, Den? U: I live in Chiba. S: I see. Chiba. Chiba is great. I want to go there. It is famous for Tokyo Disney Land, isn't it? U: Yeah, although it is not regarded as part of Chiba. S: Oh, really. I was wondering how it's like because my friends tell me that it is great. Can you recommend me some sightseeing spots in Chiba? U: There is no great place in Chiba.
teamzunko	S: Hi, good morning. My name is Zunko. I'm promoting Zunda rice cakes around the world. Lately, I'm studying cooking. What is your favorite food? U: I like omelet-rice (rice wrapped in an omelet). S: Oh, it is western-style food, isn't it? It actually originated from Japan, did you know that? U: Oh, it is Japanese food, then. I didn't know that. S: Talking about Japanese food, do you know Nikiri? It is soy sauce evaporated with sake or sweet sake. It is widely used in Japanese dishes. U: Oh, yes. You have to evaporate alcohol when you use sweet sake.

Fig. 2 Dialogue excerpts from live event. S and U indicate system and user, respectively. The dialogues were originally in Japanese and were translated by the authors

Fig. 3 Photos of live event. Designated speaker talking to chatbot on Telegram (left) and oral presentation by a participating team (right)

utterances. The system has the personality of the character, Zunko Tohoku,[7] and uses rules to produce character-like utterances. If the rules do not match, a seq2seq model is used to generate an utterance.

We now describe the procedure and results of the live event. We also discuss the analysis of the questionnaire responses before and after the live event, from which we identified the problems with current chat-oriented dialogue systems. See Fig. 3 for the photos of the event.

4.1 Procedure

The procedure was as follows:

1. Each system was to talk with a designated speaker twice. Representatives from the three teams drew lots to decide the order of the systems. Since we had three systems, six dialogues were to be performed. The order was decided to be: NTTCS, NTTdocomo, teamzunko, NTTdocomo, teamzunko, and NTTCS.
2. Three of the four designated speakers (A–D) drew lots to decide to which system they would talk. The speakers were selected from senior researchers who are experts in dialogue systems. One of the speakers (A), who did not draw lots, talked to all three systems, while B–D were assigned to one of the systems.
3. Before the dialogues, the audience filled out a questionnaire,[8] which asked what they think are the current problems with chat-oriented dialogue systems. We prepared a list of 30 problems, and the audience was requested to choose the five most important from the list. See Table 4 for the full list of problems.
4. The six dialogues were performed by repeating the following process six times:

[7] https://zunko.jp/.

[8] We used Google Form for the questionnaire and the following evaluations.

Table 4 List of problems from questionnaire regarding chat-oriented dialogue systems. Authors solicited problems in advance from dialogue research community in Japan and manually clustered them into 30 problems. Category indicates label used in clustering

No.	Category	Description of problem
1	Objective	The objective of the dialogue is unclear
2	Objective	The intention of the system is unclear
3	Objective	The dialogue is not informative
4	Objective	The dialogue is not interesting/fun
5	Topics	The system cannot cope with many topics
6	Topics	The system cannot shift to desired topics
7	Knowledge	The system lacks common-sense knowledge
8	Knowledge	The system lacks sociality
9	Knowledge	The system lacks awareness of the environment/situation
10	Knowledge	The system delivers incorrect information
11	Knowledge	The system delivers outdated information
12	User adaptation/Personality	The system does not adapt to the user
13	User adaptation/Personality	The relationship between the system and user does not change
14	User adaptation/Personality	The system does not make use of dialogue history
15	User adaptation/Personality	The system lacks personality
16	Understanding	The level of understanding is poor/shallow
17	Understanding	The system does not understand emotion
18	Understanding	The system does not understand/answer questions
19	Generation	The system generates ill-formed/ungrammatical sentences
20	Generation	The underlying intention of the utterance is unclear
21	Generation	The utterance does not reflect the dialogue context
22	Generation	The system repeats the same or similar utterance
23	Generation	The system generates contradicting utterances
24	Miscellaneous	The system cannot adequately grab/release initiatives
25	Miscellaneous	The system cannot recover from errors
26	Miscellaneous	It is difficult to determine what one can say to the system
27	Miscellaneous	The system cannot cope with multi-modal input
28	Business/Deployment	The implementation cost is high
29	Business/Deployment	The business model has not yet been established
30	Business/Deployment	It is difficult to evaluate the dialogue quality

a. A designated speaker talked to his/her assigned bot on the Telegram interface while the audience watched the interaction.
b. After the dialogue, the audience (except for those with conflict of interest) rated each utterance of the dialogue with one of three labels (good, bad, or ambivalent) before finally providing a single rating about their willingness to

talk to the system in question on a five-point Likert scale (as in the preliminary selection, 1 is best and 5 worst).

5. The participating teams presented their systems through oral presentations. They also answered questions from the audience regarding the behavior of their systems. The dialogue logs were displayed on the screen so that utterance-level discussions could be done.
6. The audience filled out the same questionnaire, selecting the five most important problems after watching/evaluating the dialogues and listening to the presentations.

4.2 Results

Table 5 shows the results of the live event. NTTCS achieved the best averaged score of 2.81, followed by NTTdocomo (3.06) and teamzunko (3.48). Different from the preliminary selection, NTTCS was rated the best. It is also clear that the range of the scores differs with those of the crowd-workers (see Table 3). The audience seemed to have rated the systems more severely, although it may be because the raters and speakers were different in the live event.

When we looked at the comments given for each system, we found that NTTCS was favored because of the natural dialogue flow resulting from constraining the topic to travel, although there was criticism that the dialogue was too system-initiative and monotonous. NTTdocomo's system was evaluated highly for its friendly character speaking in the Kansai (Osaka area of Japan) dialect. The downside was its abrupt topic-shifts and its failure to understand open-domain utterances. For teamzunko, the audience liked the variety of utterances made from its different sources, although there was criticism about the frequent dialogue breakdowns due to their system ignor-

Table 5 Results from live event. Average indicates score averaged over all ratings given to particular system

Rank	Team	Dialogue	Averaged rating	No. of raters
1	NTTCS	1st dialogue	2.69	97
		2nd dialogue	2.93	99
		Average	**2.81**	
2	NTTdocomo	1st dialogue	3.18	101
		2nd dialogue	2.95	101
		Average	**3.06**	
3	teamzunko	1st dialogue	3.58	92
		2nd dialogue	3.40	98
		Average	**3.48**	

ing user utterances and producing ungrammatical utterances from seq2seq models. Overall, NTTCS's system was perceived the best probably because it was keeping track of the dialogue by restricting the topic, while others had difficulty in dealing with open-domain utterances of users, which is still fraught with problems.

4.3 Analysis of Questionnaire

Table 6 shows the top ten problems after the live event and the ranking of these problems before the live event so that we can compare the changes in ranking. After the audience witnessed the dialogues, the rankings significantly changed; the understanding of user utterances, recognition of the topics, and capability of answering questions now occupy the top three, indicating that the fundamental function of understanding the meaning of utterances is still a challenge in chat-oriented dialogue systems. As the changes in the ranking suggest, the event successfully highlighted the difficulty with current chat-oriented dialogue systems.

5 Summary and Future Work

We described a competition entitled "the dialogue system live competition" we organized in which the audience, consisting mainly of researchers in dialogue systems, watched and evaluated a live dialogue conducted between users and dialogue systems. The motivation behind the competition was to cultivate state-of-the-art techniques in dialogue systems and enable the dialogue community to share the problems with current dialogue systems.

We had a total of 12 entries, out of which 11 systems were evaluated during the preliminary selection. Three systems proceeded to the live event and were evaluated by approximately 100 people. Below are the findings from the competition:

- The combination of rule- and generation-based approaches seems appropriate to produce good dialogues. The use of multiple knowledge sources also helps. Currently, neural-based methods alone do not lead to high-quality dialogues.
- With the current level of dialogue technology, it greatly helps to constrain the topic of conversation (such as travel) to keep the dialogue on track.
- The live event successfully identified the difficulty with current chat-oriented dialogue systems. After the event, the problems perceived to be important were mostly regarding the fundamental understanding functions of a system, such as understanding of utterances, recognition of the topics, and question answering.

We believe our findings are general enough because we used high-quality dialogue systems that are currently available, although it may be possible that the problems we identified may be specific to the systems of the live event. We want to further

Table 6 Ten most important problems after live event. Previous ranking indicates problem ranking from questionnaire before live event

Rank	Problem description	Previous ranking	Change in ranking
1	The level of understanding is poor/shallow	2	UP
2	The system cannot shift to desired topics	8	UP
3	The system does not understand/answer questions	13	UP
4	The utterance does not reflect the dialogue context	1	DOWN
5	The intention of the system is unclear	18	UP
6	The system does not make use of dialogue history	11	UP
7	The objective of the dialogue is unclear	7	SAME
8	The system repeats the same or similar utterance	19	UP
9	The system does not adapt to the user	6	DOWN
10	The system cannot cope with many topics	5	DOWN

clarify the problems and pursue state-of-the-art methods in chat-oriented dialogue systems by holding another live competition.

Acknowledgements The authors would like to thank the Japanese society for artificial intelligence (JSAI) for financially supporting the competition. We also thank the teams who participated in the competition and those who evaluated the dialogues during the live event.

References

1. Abe K, Sato S, Sato T, Fujii R, Matsuda K, Suzuki M, Yamaguchi K, Akama R, Ouchi H, Suzuki J, Inui K (2018) Zunkobot: a multiple knowledge bases integrated chatbot system. In: Proceedings of JSAI SIG-SLUD, pp 112–117. (In Japanese)
2. Black AW, Eskenazi M (2009) The spoken dialogue challenge. In: Proceedings of SIGDIAL, pp 337–340

3. Higashinaka R, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y, Mizukami M (2015) Towards taxonomy of errors in chat-oriented dialogue systems. In: Proceedings of SIGDIAL, pp 87–95
4. Higashinaka R, Funakoshi K, Kobayashi Y, Inaba M (2016) The dialogue breakdown detection challenge: task description, datasets, and evaluation metrics. In: Proceedings of LREC, pp 3146–3150
5. Ritter A, Cherry C, Dolan WB (2011) Data-driven response generation in social media. In: Proceedings of EMNLP, pp 583–593
6. Shang L, Sakai T, Lu Z, Li H, Higashinaka R, Miyao Y (2016) Overview of the NTCIR-12 short text conversation task. In: Proceedings of NTCIR
7. Sugiyama H, Narimatsu H, Mizukami M, Arimoto T (2018) Empirical study on domain-specific conversational dialogue system based on context-aware utterance understanding and generation. In: Proceedings of JSAI SIG-SLUD, pp 118–123. (In Japanese)
8. Tsunomori Y, Onishi K, Fujimoto H, Kadono K, Yoshimura T, Isoda, Y (2018) Development of a customizable open domain chat-oriented dialogue system. In: Proceedings of JSAI SIG-SLUD, pp 124–127. (In Japanese)
9. Williams J, Raux A, Ramachandran D, Black A (2013) The dialog state tracking challenge. In: Proceedings of SIGDIAL, pp 404–413

Multimodal Dialogue Data Collection and Analysis of Annotation Disagreement

Kazunori Komatani, Shogo Okada, Haruto Nishimoto, Masahiro Araki, and Mikio Nakano

Abstract We have been collecting multimodal dialogue data [1] to contribute to the development of multimodal dialogue systems that can take a user's non-verbal behaviors into consideration. We recruited 30 participants from the general public whose ages ranged from 20 to 50 and genders were almost balanced. The consent form to be filled in by the participants was updated to enable data distribution to researchers as long as it is used for research purposes. After the data collection, eight annotators were divided into three groups and assigned labels representing how much a participant looks interested in the current topic to every exchange. The labels given among the annotators do not always agree as they depend on subjective impressions. We also analyzed the disagreement among annotators and temporal changes of impressions of the same annotators.

K. Komatani (✉) · H. Nishimoto
Osaka University, Osaka, Japan
e-mail: komatani@sanken.osaka-u.ac.jp

H. Nishimoto
e-mail: nishimoto@ei.sanken.osaka-u.ac.jp

S. Okada
Japan Advanced Institute of Science and Technology, Ishikawa, Japan
e-mail: okada-s@jaist.ac.jp

RIKEN AIP, Tokyo, Japan

M. Araki
Kyoto Institute of Technology, Kyoto, Japan
e-mail: araki@kit.jp

M. Nakano
Honda Research Institute Japan Co., Ltd, Saitama, Japan
e-mail: nakano@jp.honda-ri.com

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_17

1 Introduction

Dialogues consist of various elements including verbal information conveyed by spoken languages and non-verbal information. Multimodal information including such non-verbal information is important when constructing various systems such as dialogue robots and empathetic agents. It is desirable that such a system can detect intention, attitude, and emotion not necessarily expressed through spoken language.

We have been constructing a multimodal corpus of human-system multimodal dialogues that can be distributed to researchers, which will contribute to the development of multimodal dialogue systems. Since 2016, we have performed a number of activities to build such a shared corpus,[1] which will enable us to analyze user behaviors and contribute to improving components of multimodal dialogue systems.

We focus on analyzing how humans behave toward multimodal dialogue systems rather than analyzing human-human dialogues. While a number of shared corpora of multimodal dialogues among humans have been built and there have been plenty of studies analyzing these corpora (e.g., [2, 10]), user behaviors in human-system dialogues are different from those in human-human dialogues especially when the users realize that they are talking to a system.

This paper reports our new data collection of multimodal dialogues and the results of its annotation disagreement analyses. After outlining related work in Sect. 2, we describe the details of the data collection and annotation in Sect. 3. In Sects. 4 and 5, we show the results of several analyses for annotation disagreement. In Sect. 4, we present a model to show how each annotator's results are deviated. In Sect. 5, we show how the annotation results changed when annotators assigned the labels again about one month after the first annotation. We conclude this paper by showing our perspective and issues in sharing such corpora in Sect. 6.

2 Related Work

Several meeting conversation corpora with multiple participants have been released and shared for analyzing human-human multimodal conversation, such as the Augmented Multi-party Interaction (AMI) corpus [2] and the ICSI meeting corpus [10]. The Computers in Human Interaction Loop (CHIL) corpus treats human-human interactions in offices and classrooms [21], and the Video Analysis and Content Extraction (VACE) corpus also treats human-human interactions in battle-game sessions in the air force [3]. A corpus of political debates in a TV program has also been shared for analyzing social interactions [20].

Several multimodal corpora, which are not those of interactions, have also been published, such as ones for assessing public speaking ability and anxiety [5] and for

[1] These activities are being conducted by a working group (Human-System Multimodal Dialogue Sharing Corpus Building Group) under SIG-SLUD of the Japanese Society for Artificial Intelligence (JSAI).

recognizing group-level emotion on in-the-wild data [6]. Such multimodal analyses have been used for constructing systems; for example, a system that analyzes non-verbal human behaviors was developed and used to assess indicators of psychological distress, such as depression and post-traumatic stress disorder (PTSD) [18].

Our target is to construct a human-system dialogue corpus. The most important difference between human-human and human-system dialogues is whether users realize that they are talking to a system. User behaviors differ when they talk to a system or a human. Data on human-system dialogues contain real behaviors of users, which can be used to detect user states in actual dialogue systems.

Our goal in this project is to contribute to the development of multimodal dialogue systems. The corpora can be used to detect users' interests [4, 8] and boredom [16], which can lead to adaptive dialogue systems that change their question strategy and topics to talk with. Collecting and sharing such dialogue corpora can lead to the development of component techniques required in dialogue systems, which will not easily bore users by considering their states.

In terms of sharing the human-system dialogue corpus, there was a project to collect and share a text chat corpus, which also conducted shared tasks using it [7]. A competition for speech-input chatbots, the Amazon Alexa Challenge,[2] was also conducted. The target of our project is not text-input or speech-input chatbots but multimodal dialogue systems. Privacy issues for the multimodal data need to be considered because it contains participants' faces.

There have been several studies that exploit labels given by multiple annotators. Ozkan et al. used the labels obtained via crowdsourcing for predicting backchannels by integrating the classifiers for each crowdworker by using latent variables [14, 15]. Kumano et al. proposed a probabilistic model that considers an observer's (i.e. an annotator's) perception tendency [11]. Inoue et al. also proposed a latent character model of annotators to estimate a listener's engagement [9]. We analyze the disagreement of the annotation results in our corpus and also empirically confirm annotators' tendency.

3 Collecting Dialogue Data

We collected the data of human behaviors when participants talked with a system operated using the Wizard of Oz (WoZ) method. We updated the consent form to enable data distribution for research purposes and recruited various participants from the general public. After the data collection, we annotated the interest level of the participants to every exchange, similarly to our previous work [1]. The annotation was applied to every exchange, and can be used as reference labels for constructing a system that can adapt to user multimodal behaviors in each exchange.

[2] https://developer.amazon.com/alexaprize.

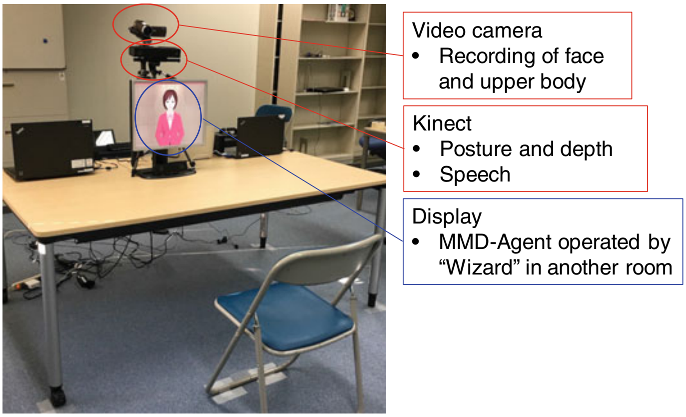


Fig. 1 Environment of data collection

3.1 System

We used a virtual agent, MMDAgent,[3] as the interface with which participants interacted. The environmental setup is shown in Fig. 1. The participants talked with the agent shown on the display.

The behavior of the participants was recorded using a video camera and a Microsoft Kinect V2 sensor, which were set up above the display. The video camera recorded the face and upper body of the participants at 30 frames per second (fps). The audio recording sampling rate was 44.1 kHz. The Kinect sensor recorded voice, depth information, and posture of their upper body. The posture information was recorded as joint positions of the participant, whose coordinate origin was the center of the infrared (IR) sensor, at 30 fps. The voice was recorded as 16 kHz WAV files. We asked the participant to clap once before the dialogue to provide a reference point to align the camera and the Kinect sensor manually.

The virtual agent was manipulated by an operator, called the "Wizard," via WoZ method. The operator simulated a dialogue system that interacts with the participants. The operator's interface is shown in Fig. 2. The operator can select a topic, utterances in the topic, and general responses used in chat dialogues, and can partly control facial expressions and actions of the virtual agent when it speaks by putting some special character sequences at the end of utterances.

[3]http://www.mmdagent.jp/.

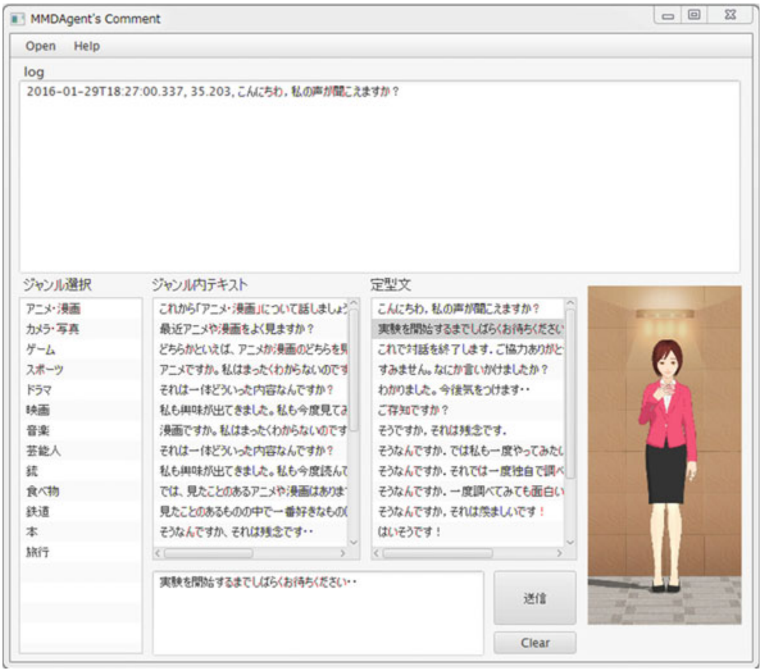


Fig. 2 GUI for Wizard [19]

3.2 *Task*

No specific task was set in the dialogue; that is, it was a chat dialogue. Before the dialogue, each participant filled out a questionnaire selecting three favorite and three non-favorite topics out of 12 topics prepared beforehand. The operator used the six chosen topics in random order, and selected questions and responses prepared in the Wizard interface after the participants spoke. The number of exchanges in each topic was about 15. An example dialogue is shown in Fig. 3 (translated from Japanese).

3.3 *Participants and Collected Data*

The participants comprised 30 males and females, whose ages ranged from 20 to 50, recruited from the general public through a recruitment agency. We explained to them the objectives of this data collection and their rights (e.g., they can withdraw their participation in the data collection at any time) and asked them to fill in the consent form only if they agreed with the conditions. We clearly stated that the collected

```
----------------------------------------------------
S: Let’s talk about food.
U: Okay.
S: Do you like sweets?
U: Sweets, ... well, not so much.
S: Then let’s talk about cooking.
U: Okay.
S: Which do you like better, eating or cooking?
U: Well, I much more prefer to eat.
S: You like to eat. Can you recommend any food?
----------------------------------------------------
```

S and U denote the system (Wizard) and user (participant) utterances, respectively.

Fig. 3 Example WoZ dialogue

data can be distributed to researchers for research and development purposes on the premise that the researchers submit a written oath on use. In addition, explicit permission was asked of the participants for use of their facial images in conference presentations and technical papers.

We collected data from 29 participants; the data of one participant could not be recorded due to machine trouble. The duration of the data is about 15 min per participant. Dialogue data with each participant contains about 80 exchanges for the six topics.

3.4 Annotation

We annotated labels that denote an interest level, i.e., whether a participant seemed to be interested in the current dialogue topic. The labels basically consist of three choices: interest (o), unknown (t), and no interest (x). If a system error occurred, error (e) can also be given. The interest level is a different notion from engagement [13, 17]. We believed the engagement was almost always established in such situations because these were dyadic interactions without any distracting factors, so we focused on the interest level, which can be helpful for constructing multimodal dialogue systems that select the next topic based on its estimation results.

The labels were given to each exchange (i.e., a pair of a system utterance and a user utterance). More specifically, an exchange begins from the start time of a system utterance and ends at the start time of its next system utterance.

Annotators were instructed to consider various features of the participants, such as facial expressions, prosody, utterance content, etc., and not to determine the labels only with a specific modality. They were also instructed not to give labels only from a part of the exchange, and to give labels considering differences among individual participants after watching the entire recording of the target participant.

Table 1 Amount of data and annotator groups

Group	Num. participants	Num. exchanges	Num. annotators
A	10	844	6
B	10	816	3
C	9	762	3

A total of eight annotators[4] were assigned to three groups, as shown in Table 1. We did not use crowdsourcing because we need to protect the participants' personal information. First, three annotators were assigned to groups B and C, respectively, so that we can use a majority vote of labels if necessary. The remaining annotators were assigned to group A so as to enable analysis of the annotation tendency by multiple annotators.

4 Analysis on Disagreement of Annotation Results

Since perception of the interest level is subjective, the labels given by the annotators do not completely agree. The inter-rater agreement scores (Fleiss' κ coefficients) among annotators in each group are shown in Table 2. According to [12], κ values between 0.41 and 0.60 are regarded as moderate agreement but not perfect agreement.

In this section, we analyze the annotation tendency to discuss the disagreement among labels given by annotators. First, we give examples to show our insights and formulation. We then show the tendency of all annotators in each group.

4.1 Example in Arbitrary Pair of Annotators

Table 3 shows an example confusion matrix, in which each value denotes counts of labels given by two annotators (here, B2 and B3 in group B). Each row of the matrix represents the labels given by annotator B2 while each column represents those by B3. Diagonal components in the matrix denote counts of cases where the two annotators gave the same labels, i.e., their judgments agreed. However, the count

Table 2 Inter-rater agreement scores (Fleiss' kappa coefficient) among annotators in each group

Group	κ
A	0.49
B	0.55
C	0.45

[4]Two annotators gave labels to the data of all three groups.

Table 3 Example confusion matrix of labels annotated by B2 and B3 (each row represents labels annotated by B2; each column represents those by B3.)

B2 \ B3	o	t	x	Total
o	192	19	18	229
t	23	64	9	96
x	125	68	297	490
Total	340	151	324	815

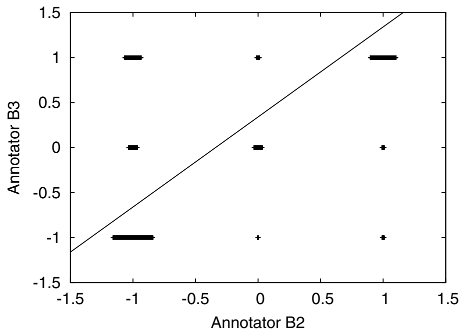


Fig. 4 Example of regression analysis (a pair of annotators B2 and B3). Since multiple pairs overlap on the nine points, we shifted plot positions of the points depending on the number of pairs

of disagreed labels shown in the lower triangular matrix ($23 + 125 + 68 = 216$) was larger than that on the upper triangular matrix ($19 + 18 + 9 = 46$). This means that B3 tended to give more "interest (o)" labels to exchanges than B2 did.

To analyze the tendency of the disagreement, we conducted a regression analysis based on the function $y = x + b$, which is a parallel translation of x to y. b in the regression model corresponds to the difference between x and y. For the regression analysis, we converted the three labels (categorical data) to continuous values (numerical data) corresponding to the three levels. Assuming the unknown (t) label has the medium value between interest (o) and no interest (x), we converted the labels x, t, o to continuous scores $-1, 0, 1$, respectively.

Let $s_{i,j} \in \{-1, 0, 1\}$ be scores of each exchange j given by annotator i. We conducted a regression analysis between all pairs of score vectors given by two annotators who belonged to the same groups (A-C). On the assumption that there is difference of scoring tendency between annotators, b is optimized so as to minimize the sum of the square errors between $s_{i_1,j}$ and $s_{i_2,j}$ (i.e., $s_{i_1,j} + b$). The sign of the estimated b indicates whether the annotator tends to annotate a higher score ("interest (o)") or lower ("no interest (x)").

Figure 4 shows the result of the regression analysis between annotators B2 and B3 as an example. Each point denotes the pair of scores ($s_{B2,j}$ and $s_{B3,j}$). Note that nine points in this figure correspond to the pairs (3×3) of scores ($s_{i,j} \in \{-1, 0, 1\}$).

After optimizing the regression model, the bias value b in the regression function (the amount of the parallel translation) was estimated as 0.26. This result shows that

$s_{B3,*}$ tends to be larger than $s_{B2,*}$. The result of the regression modeling enables us to effectively analyze the difference of scoring tendencies between annotators.

The evidence that B3 tended to annotate higher scores is also obtained using a paired sample t-test between B2 and B3. The t-test was conducted to determine whether the mean of differences between the two score sets of samples (exchanges) was zero. As a result of the t-test, the p-value was less than 10^{-27}, indicating that the average of the differences between the score sets was not zero with statistical significance.

4.2 Tendency Among All Annotators in Each Group

Next, we compare the annotation tendency among all annotators by using the regression analysis. For this purpose, we calculated average scores ($\overline{s}_j$) of all annotators belonging to each group and conducted the regression analysis between scores $s_{i,j}$ of an annotator and the averages $\overline{s}_j$. The average scores of each group were used as standards for obtaining relative relationships of each annotator's results. As an example, we show the results of the regression analysis that converts the scores of C3 to the average scores of group C in Fig. 5.

Through the regression analysis between the scores and the average scores, we can compare scoring tendencies by annotator i by ranking the bias b_i of the regression model. Table 4 shows the comparison of the estimated b_i via regression analysis in each group.

From Table 4, the result of the regression analysis shows that the biases of annotators A1, A2 and A3 were positive ($b_i > 0$) and those of A6, A5, and A4 were negative. The results indicate that annotators A1, A2, and A3 tended to give lower scores ("no-interest") than the average in Group A and, inversely, annotators A4, A5, and A6 tended to give higher scores ("interest"). In groups B and C, the biases of annotators B3 and C3 were negative ($b_i < 0$), which indicates they tended to give higher scores ("interest") than the average and when compared to the other annotators in the group.

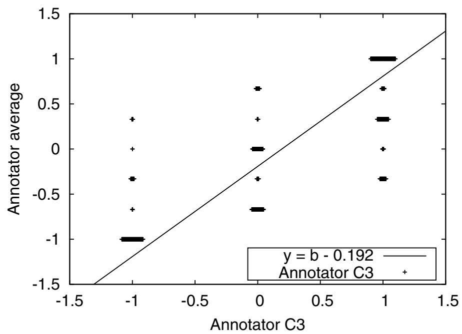


Fig. 5 Example of regression analysis using the averages of each group: scores of annotator C3 and the average scores of all annotators belonging to group C

Table 4 Bias b_i in the regression model trained by using score vectors of annotator i so as to output the average scores of all annotators (including the annotator) in each group. The larger b_i denotes larger differences between annotator scores and average scores in each group

A	b_i	B	b_i	C	b_i
A2	0.178	B2	0.138	C2	0.117
A1	0.104	B1	0.063	C1	0.076
A3	0.046	B3	−0.202	C3	−0.192
A6	−0.093				
A5	−0.094				
A4	−0.142				

The regression analysis revealed the difference of scoring tendencies between annotators. If there is significant ordinal association between the score sets given by two annotators, it is suggested that the subjective difference of scores between annotators is explainable through the regression analysis.

5 Analysis of Temporal Changes of the Same Annotators' Results

We have confirmed in the previous section that there are deviations in the annotation results depending on the individual annotators. In this section, we verify whether such deviations are caused by the annotators' characteristics or their capriciousness.

We asked two annotators in Group A (A4 and A5) to annotate again with the same data about one month after the first annotation. We then compared the first and second annotation results by κ values in addition to those with other annotators' results. The results are shown in Table 5.

Table 5 κ values between 1st and 2nd annotations in Group A

		A4(1st)	A5(1st)
1st	A1	0.426	0.564
	A2	0.463	0.536
	A3	0.438	0.613
	A4	–	0.509
	A5	0.509	–
	A6	0.473	0.438
2nd	A4	0.605	0.502
	A5	0.549	0.689

The highest κ values for A4(1st) and A5(1st) were obtained with A4(2nd) and A5(2nd), respectively, whose κ values were 0.605 and 0.689. This result indicates that the annotation results most agreed with those by the annotators themselves after one month passed, compared with those with other annotators, although the results did not match perfectly even with the same annotators. This suggests that there is an annotator-specific tendency that is preserved even after a lapse in time. Although the annotation results do not always agree because of the subjectivity of each annotator, we showed there is a certain tendency specific to each annotator, and confirmed quantitatively that it is different from other annotators.

6 Issues and Future Work

The collected multimodal dialogue data includes the personal information of participants such as their faces and voices. Therefore, the data needs to be handled carefully so as not to violate privacy issues. To keep continuous and consistent handling of such data, we are now in discussion with an organization whose mission is to distribute such corpora.

Organization distributing such data may require that the data collection procedure also be certified by its ethics review committee in addition to the collection phase. In an ethics review, it is better to write the content of the consent forms more concretely to relieve participants. On the other hand, if the usage description is very limited, opportunities to use the data will also be limited when it is shared with various researchers. It is also an important factor that researchers can easily obtain and use the data. Thus, before starting a data collection that includes personal information for the purpose of sharing, it is necessary to design in advance its consent form that balances ease of data distribution and personal information protection.

Upon request, the collected data is currently being distributed by the first author.

Acknowledgements We thank the working group members who contributed to the annotation. Ms. Sayaka Tomimasu contributed to this project during the data collection. This work was partly supported by the Research Program of "Dynamic Alliance for Open Innovation Bridging Human, Environment and Materials" in Network Joint Research Center for Materials and Devices.

References

1. Araki M, Tomimasu S, Nakano M, Komatani K, Okada S, Fujie S, Sugiyama H (2018) Collection of multimodal dialog data and analysis of the result of annotation of users' interest level. In: Proceedings of international conference on language resources and evaluation (LREC)
2. Carletta J (2007) Unleashing the killer corpus: experiences in creating the multi-everything AMI meeting corpus. Lang Resour Eval 41(2):181–190

3. Chen L, Rose RT, Qiao Y, Kimbara I, Parrill F, Welji H, Han TX, Tu J, Huang Z, Harper M, Quek F, Xiong Y, McNeill D, Tuttle R, Huang T (2006) VACE multimodal meeting corpus. In: Proceedings of the 2nd international conference on machine learning for multimodal interaction (MLMI05), pp 40–51. https://doi.org/10.1007/11677482_4
4. Chiba Y, Ito M, Nose T, Ito A (2014) User modeling by using bag-of-behaviors for building a dialog system sensitive to the interlocutor's internal state. In: Proceedings of annual meeting of the special interest group on discourse and dialogue (SIGDIAL), pp 74–78. http://www.aclweb.org/anthology/W14-4310
5. Chollet M, Prendinger H, Scherer S (2016) Native vs. non-native language fluency implications on multimodal interaction for interpersonal skills training. In: Proceedings of international conference on multimodal interaction (ICMI), pp 386–393. http://doi.acm.org/10.1145/2993148.2993196
6. Dhall A, Goecke R, Ghosh S, Joshi J, Hoey J, Gedeon T (2017) From individual to group-level emotion recognition: EmotiW 5.0. In: Proceedings of international conference on multimodal interaction (ICMI). ACM, New York, NY, USA, pp 524–528. http://doi.acm.org/10.1145/3136755.3143004
7. Higashinaka R, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y, Mizukami M (2015) Towards taxonomy of errors in chat-oriented dialogue systems. In: Proceedings of annual meeting of the special interest group on discourse and dialogue (SIGDIAL), pp 87–95
8. Hirayama T, Sumi Y, Kawahara T, Matsuyama T (2011) Info-concierge: proactive multi-modal interaction through mind probing. In: The Asia Pacific signal and information processing association annual summit and conference (APSIPA ASC 2011)
9. Inoue K, Lala D, Takanashi K, Kawahara T (2018) Latent character model for engagement recognition based on multimodal behaviors. In: Proceedings of international workshop on spoken dialogue systems (IWSDS)
10. Janin A, Baron D, Edwards J, Ellis D, Gelbart D, Morgan N, Peskin B, Pfau T, Shriberg E, Stolcke A, Wooters C (2003) The ICSI meeting corpus. In: Proceedings of IEEE international conference on acoustics, speech & signal processing (ICASSP), pp I–364–I–367. https://doi.org/10.1109/ICASSP.2003.1198793
11. Kumano S, Otsuka K, Matsuda M, Ishii R, Yamato J (2013) Using a probabilistic topic model to link observers' perception tendency to personality. In: Proceedings of the ACM conference on affective computing and intelligent interaction (ACII), pp 588–593. https://doi.org/10.1109/ACII.2013.103
12. Landis JR, Koch GG (1977) The measurement of observer agreement for categorical data. Biometrics 33(1):159–174
13. Nakano YI, Ishii R (2010) Estimating user's engagement from eye-gaze behaviors in human-agent conversations. In: Proceedings of international conference on intelligent user interfaces (IUI), pp 139–148. https://doi.org/10.1145/1719970.1719990
14. Ozkan D, Morency LP (2011) Modeling wisdom of crowds using latent mixture of discriminative experts. In: Proceedings of annual meeting of the association for computational linguistics (ACL): human language technologies (HLT), pp 335–340. http://dl.acm.org/citation.cfm?id=2002736.2002806
15. Ozkan D, Sagae K, Morency L (2010) Latent mixture of discriminative experts for multimodal prediction modeling. In: Proceedings of international conference on computational linguistics (COLING), pp 860–868. http://aclweb.org/anthology/C10-1097
16. Shibasaki Y, Funakoshi K, Shinoda K (2017) Boredom recognition based on users' spontaneous behaviors in multiparty human-robot interactions. In: Proceedings of multimedia modeling, pp 677–689. https://doi.org/10.1007/978-3-319-51811-4_55
17. Sidner C, Kidd C, Lee C, Lesh N (2004) Where to look: a study of human-robot engagement. In: Proceedings of international conference on intelligent user interfaces (IUI), pp 78–84. https://doi.org/10.1145/964442.964458
18. Stratou G, Morency LP (2017) Multisense—context-aware nonverbal behavior analysis framework: a psychological distress use case. IEEE Trans Affect Comput 8(2):190–203. https://doi.org/10.1109/TAFFC.2016.2614300

19. Tomimasu S, Araki M (2016) Assessment of users' interests in multimodal dialog based on exchange unit. In: Proceedings of the workshop on multimodal analyses enabling artificial agents in human-machine interaction (MA3HMI'16). ACM, New York, NY, USA, pp 33–37. http://doi.acm.org/10.1145/3011263.3011269
20. Vinciarelli A, Dielmann A, Favre S, Salamin H (2009) Canal9: a database of political debates for analysis of social interactions. In: 2009 3rd international conference on affective computing and intelligent interaction and workshops, pp 1–4. https://doi.org/10.1109/ACII.2009.5349466
21. Waibel A, Stiefelhagen R (2009) Computers in the human interaction loop, 1st edn. Springer Publishing Company, Incorporated, Berlin

Analyzing How a Talk Show Host Performs Follow-Up Questions for Developing an Interview Agent

Hiromi Narimatsu, Ryuichiro Higashinaka, Hiroaki Sugiyama, Masahiro Mizukami, and Tsunehiro Arimoto

Abstract This paper aims to reveal how human experts delve into the topics of a conversation and to apply such strategies in developing an interview agent. The purpose of recent studies on chat-oriented dialogue systems has centered around how to respond in conversation to a wide variety of topics. However, in interviews, it is also important to delve into particular topics to deepen the conversation. Since interviewers in talk shows, such as talk show hosts, are considered to be proficient at such skills, in this paper we analyze how a talk show host interviews her guests in a TV program and thus reveal the strategies used to delve into specific topics. In particular, we focus on follow-up questions and analyze the kinds of follow-up questions used. We also develop a prediction model that judges whether the next utterance should be a follow-up question and then evaluate the model's performance.

1 Introduction

Recently, interview agents have been attracting attention because they have practical uses and the resulting dialogue can be engaging. Various types of interview agents have been studied, such as SimSensei [1] for performing mental care through interviews and a job-interviewer robot [4]. However, these systems interview users in a

H. Narimatsu (✉) · R. Higashinaka · H. Sugiyama · M. Mizukami · T. Arimoto
NTT Communication Science Laboratories, 2-4 Hikaridai, Seika-cho, Kyoto, Japan
e-mail: hiromi.narimatsu.eg@hco.ntt.co.jp

R. Higashinaka
e-mail: ryuichiro.higashinaka.tp@hco.ntt.co.jp

H. Sugiyama
e-mail: hiroaki.sugiyama.kf@hco.ntt.co.jp

M. Mizukami
e-mail: masahiro.mizukami.df@hco.ntt.co.jp

T. Arimoto
e-mail: tsunehiro.arimoto.ub@hco.ntt.co.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_18

fixed setting and cannot converse with users over a wide variety of topics, as achieved in current chat-oriented dialogue systems.

This work lies at the intersection of chat-oriented dialogue systems and interview agents. While recent studies on chat-oriented dialogue systems aim to respond to a wide variety of conversation topics [3, 9], the dialogue is not very engaging because the topics change frequently, making users less engaged. We want to add interviewing skills to a chat-oriented dialogue system so that the dialogue can be more engaging and entertaining.

In order to gain insights into how to perform interviews through a chat-oriented dialogue, we analyze the behavior of a talk show host, who is considered to be proficient at such skills. In particular, we focus on follow-up questions and analyze the kinds of follow-up questions that are typically used in interviews. We also develop a prediction model to judge whether the next utterance should be a follow-up question and then evaluate this model's performance.

2 Related Work

Listening agents, interview agents, and chat-oriented dialogue systems have different ways of keeping users engaged in a conversation.

Listening agents attentively listen to users [2, 7] while making timely acknowledgments and multi-modal behaviors, which gives the users a sense of being listened to. However, they typically do not answer questions or ask questions.

Interview agents can create engaging conversations by asking questions in an interview-style [1, 2]. This strategy is effective in requesting information and in prompting users to extend their talk as long as possible. However, such an interview is not likely to be entertaining because the typical aims are to simply elicit information or perform a task-specific interview.

Recently, an abundance of work on chat-oriented dialogue systems has been reported [3, 9]. The typical strategy adopted for making users engage with such systems is to generate informative or interesting utterances [5]. Methods of generating questions have also been studied [8]; however, how to make effective questions, especially follow-up questions, within a chat-oriented dialogue has rarely been considered.

This paper adds interviewing skills to a chat-oriented dialogue system in order to produce more engaging dialogue. We assume that interviewers on TV talk shows have the required skills to produce an engaging conversation. Therefore, we analyzed the actual interviewing behaviors of a talk show host to mine the relevant strategies from the viewpoint of follow-up questions.

3 Dataset

We collected actual conversations between a professional interviewer and guests on a famous Japanese TV show "Tetsuko no Heya[1]" (meaning "Tetsuko's room" in English) that has a conversational-style format. On this TV show, the interviewer (Tetsuko Kuroyanagi) talks with a different guest on each program in a conversational style, and the audience enjoys the conversation and the information elicited from the guests. The number of dialogues in our corpus is 447, and each dialogue contains about 480 utterances.

Since attentive listening and urging strategies are particularly used in parts of the conversation, especially when the guest supplies some pieces of information on the same topic, we extracted such segments in a conversation *story*. We also defined utterance functions of listening and urging signs and assigned one of the functions to each utterance within stories.

3.1 Annotation

The definition of a story is as follows:

- It consists of utterance sequences about the same topic. Not only the story itself but also neighboring conversation blocks about the same topic are included in a story.
- The story includes at least two turns of information provision from the guest.

Then, we newly defined the utterance functions of listening and urging based on the categories of Maynard [6]. Specifically, we defined eight utterance functions related to listening and urging (See Table 1).

3.2 Statistics of the Dataset

The number of interviewer's utterances in the story is 57,333, and Fig. 1 shows the ratio of utterance functions. The ratio of FQ is highest in all interviewer utterances except for Others. This highlights the importance of FQ and indicates that the interviewer works to assist the guest in easily continuing his/her story.

To clarify the flow of transitions between the utterance functions in the interviewer, we analyzed the transition rate between each pair of functions (Fig. 2) except for *Others*. The results show that the interviewer repeats follow-up questions by sometimes inserting filler and admiration. Furthermore, Question (Ir) continuously occurs with high frequency.

[1] https://www.tv-asahi.co.jp/tetsuko/.

Table 1 Utterance functions

Utterance function	Explanation
Filler	Light expression of understanding
Repeat	Show understanding of the important words in utterances
Admiration	Show understanding that the point made is worthy of admiration
Urging	Show recognition that the current story continues
Paraphrasing	Show deeper understanding by rephrasing using other words
Question (Information request: Ir)	Questions to request new information
Follow-up Question: FQ	Questions to delve into the current topic
Others	Utterances not corresponding to any of above functions (this includes statements of impressions and opinions)

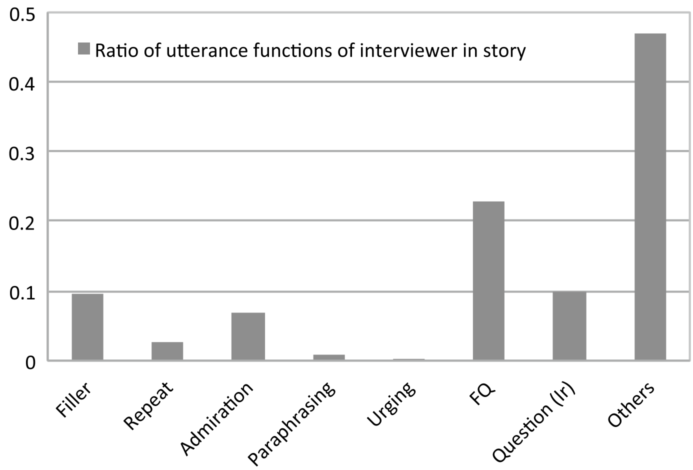


Fig. 1 Utterance function frequency rate of interviewer within the story

4 Generation of Follow-Up Questions

4.1 Judgment of Whether the Next Utterance Is a Follow-Up Question

As one of the challenges toward achieving an interviewer agent, we decided to develop a model to judge whether the next utterance is a follow-up question. The

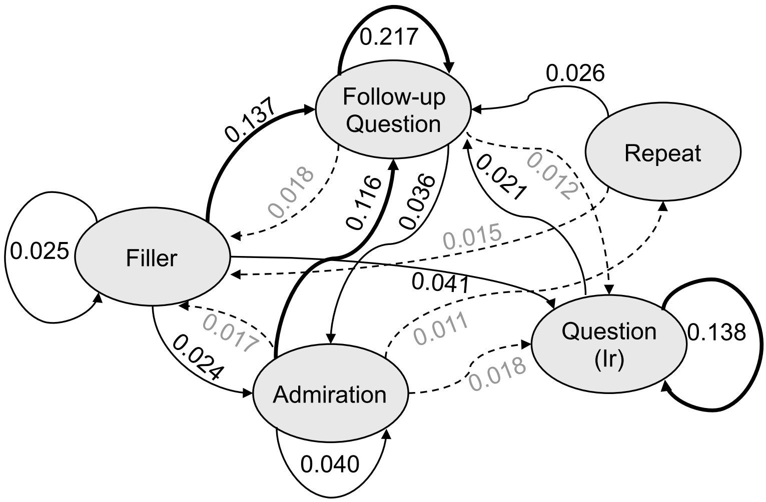


Fig. 2 Flow of transitions between utterance functions where the rate is more than 0.01

Table 2 Judgment of next follow-up question

	Pre.	Rec.	F1
DT	0.493	0.503	0.498
LR	0.609	0.433	**0.510**
SVM	0.493	0.503	0.498

number of training and test data (utterances) are 30,000 and 6,384, respectively. The number of follow-up questions in these datasets is 12,877 (42.9%) and 2,525 (39.6%), respectively.

We compared three classifier models: decision tree (DT), support vector machine (SVM), and logistic regression (LR). We first assigned dialogue-act labels to the utterances automatically using our dialogue-act classifier. Then, the dialogue-act labels in the latest n-th utterances were used as features for classification. Here, the latest n-th utterances include those of an interviewer and a guest.

Table 2 shows that LR achieved the highest F1 score with 0.510, although most of the scores are similar. To further clarify the decision process, we analyzed the trained model of the decision tree (See Fig. 3). The decision tree indicates that whether the latest guest utterance includes a Question (fact)/Question (Ir) is one of the strong features of judgment; it seems that there is a non-listening conversational turn of the interviewer when questions are included in the guest utterances.

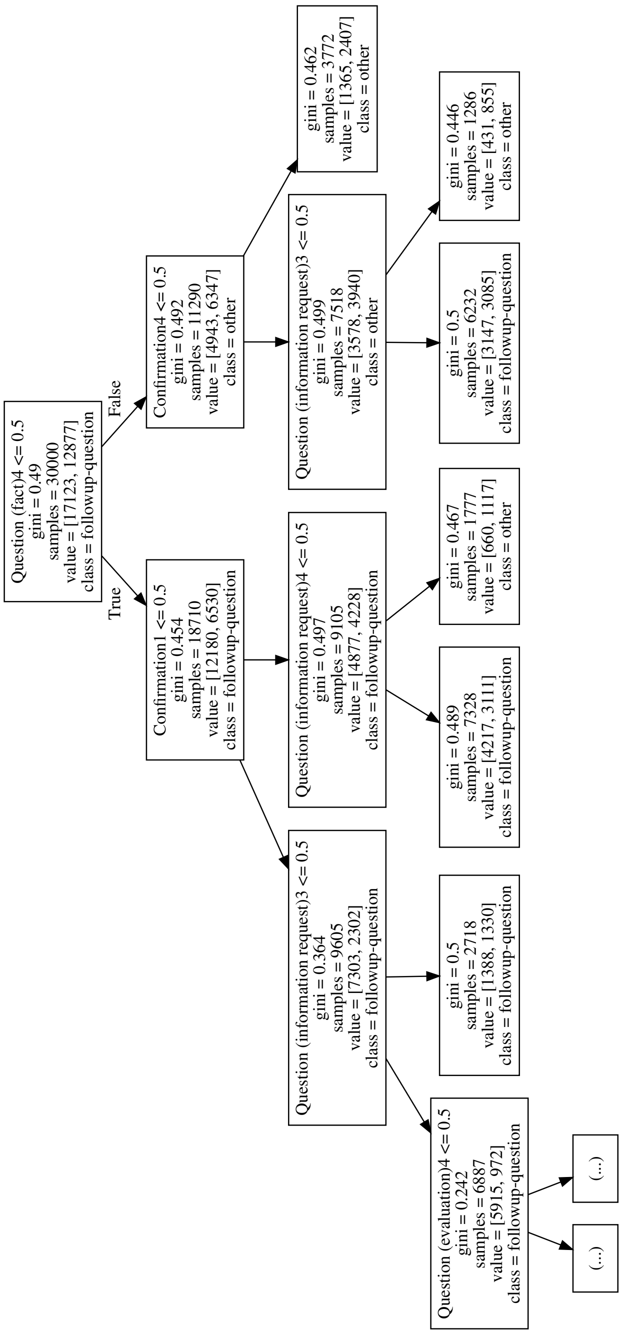


Fig. 3 Graph of trained decision tree model (depth = 3). The last number in each feature name shows the latest n-th utterance. 1 and 3 indicate the guest utterances

4.2 *Discussion for the Generation*

In order to gain insights for realizing a follow-up question generator, we analyzed the follow-up questions in more detail.

By using the estimated dialogue-act labels for the utterances, we examined the ratio of dialogue-acts in follow-up questions (Fig. 4). As a result, we found that the most frequent dialogue-acts are question (fact), confirmation, and question (information request).

Furthermore, we counted the appearance number of words in follow-up questions (Fig. 5). The number of words that appear only once is significantly large. This implies that the interviewer asked follow-up questions without using patterned questions but instead various types of representations, such as words in the context. Therefore, generating a follow-up question with certain types of dialogue-acts is important, and using the context words in the follow-up question will probably improve the quality of the follow-up questions.

Finally, we analyzed the follow-up question types by dividing them into 5W1H question categories (Table 3), indicating that "What" is the most frequent followed by "When" and "How". We also show several examples of questions along with their dialogue-acts and 5W1H categories in Table 4.

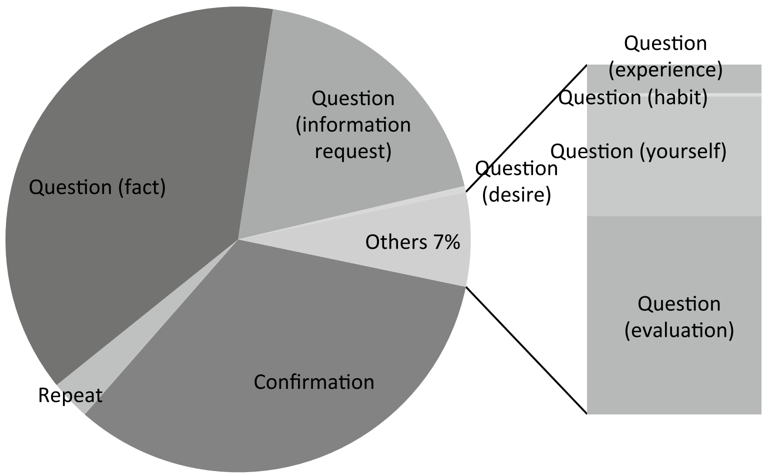


Fig. 4 Ratio of dialogue-act in follow-up question of interviewer

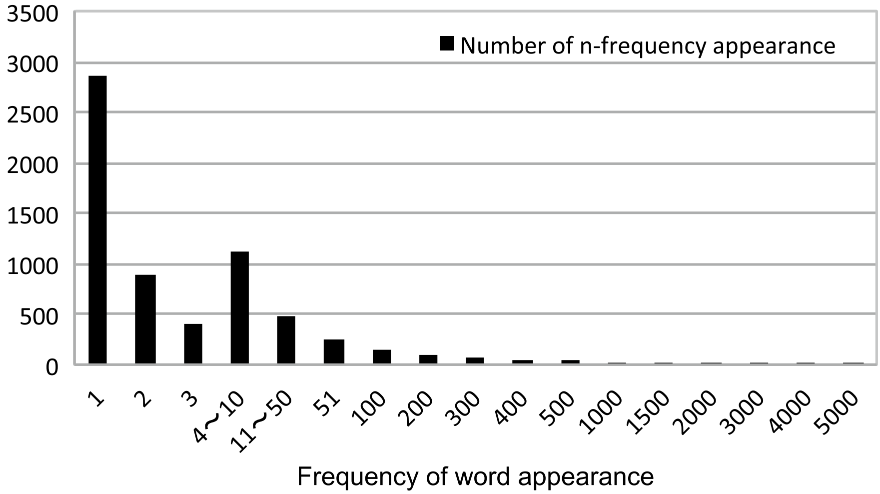


Fig. 5 Number of *n*-appearance word in follow-up question of interviewer

Table 3 Rate of 5W1H question categories in follow-up questions

	What	When	Where	Who	Why	How
Rate (%)	41.3	20.1	7.3	1.7	6.4	17.2

Table 4 Follow-up question examples

Context	Interviewer's follow-up question	Dialogue-act	5W1H category
Guest: My body got sore on stage.	Oh my god, what did you do?	Question (fact)	What
Interviewer: Well, did you leave your own shop to others? Guest: Yes.	You've been worried, haven't you?	Confirmation	Other
Guest: I ate the famous food there.	How was it? Delicious?	Question (Eval)	How

5 Conclusion

For the purpose of developing an interview agent that can delve into stories on a certain topic, we analyzed an actual TV talk show's conversation to reveal how the host interviews her guests. In particular, we focused on conversational segments in which the same topic is discussed, and we analyzed how a skilled interviewer listens to a guest and urges the guest to continue an interesting story. Our analysis shows that the high-frequency utterance function used by an interviewer in pursuing a story is the technique of follow-up questions. Through this work, we developed a simple model to judge whether the next utterance should be a follow-up question, and this model could be useful in developing an interview agent. In future work, we will analyze the specific content and utterances in detail and consider how to develop a follow-up question generator. Another study on modeling the order of dialogue-acts in a story would also be effective for creating a useful interview agent.

References

1. DeVault D, Artstein R, Benn G, Dey T, Fast E, Gainer A, Georgila K, Gratch J, Hartholt A, Lhommet M et al. (2014) Simsensei kiosk: a virtual human interviewer for healthcare decision support. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems, pp 1061–1068
2. Han S, Lee K, Lee D, Geunbae Lee G (2013) Counseling dialog system with 5w1h extraction. In: Proceedings of the SIGDIAL 2013 conference, pp 349–353
3. Higashinaka R, Imamura K, Meguro T, Miyazaki C, Sugiyama H, Hirano T, Makino T, Matsuo Y (2014) Towards an open-domain conversational system fully based on natural language processing. Proc. of COLING 2:928–939
4. Kawahara T (2018) Spoken dialogue system for a human-like conversational robot erica. In: Proceedings of IWSDS, keynote speech
5. Mashimo R, Umetani T, Kitamura T, Nadamoto A (2014) Generating funny dialogue between robots based on japanese traditional comedy entertainment. In: Proceedings of the 2014 conference on interactive entertainment. ACM, pp 1–7
6. Maynard SK (1986) On back-channel behavior in Japanese and English casual conversation. Linguistics 24(6):1079–1108
7. Meguro T, Higashinaka R, Minami Y, Dohsaka K (2010) Controlling listening-oriented dialogue using partially observable Markov decision process. In: Proceedings of COLING, pp 761–769
8. Nouri E, Artstein R, Leuski A, Traum DR (2011) Augmenting conversational characters with generated question-answer pairs. In: Proceedings of AAAI fall symposium: question generation
9. Sugiyama H, Meguro T, Higashinaka R, Minami Y (2014) Open domain utterance generation using phrase pairs based on dependency relations. In: Proceedings of spoken language technology workshop, pp 60–65

Chatbots and Conversational Agents

Chat-Oriented Dialogue System That Uses User Information Acquired Through Dialogue and Its Long-Term Evaluation

Yuiko Tsunomori, Ryuichiro Higashinaka, Takeshi Yoshimura, and Yoshinori Isoda

Abstract A chat-oriented dialogue system can become more likeable if it can remember information about users and use that information during a dialogue. We propose a chat-oriented dialogue system that can use user information acquired during a dialogue and discuss its effectiveness on long-term interaction. In our subjective evaluation over five consecutive days, we compared three systems: a system that can remember and use user information over multiple days (proposed system), one that can only remember user information within a single dialogue session, and another that does not remember any user information. We found that users were significantly more satisfied with our proposed system than with the other two. This paper is the first to verify the effectiveness of remembering on long-term interaction with a fully automated chat-oriented dialogue system.

1 Introduction

The demand for chat-oriented dialogue systems has been increasing [1, 12, 15, 18, 21]. To construct a chat-oriented dialogue system that is affective and can be used long-term by users, it is important that users and systems get to know each other well [2, 14]. Therefore, ways for dialogue systems to remember and use user information

Y. Tsunomori (✉) · T. Yoshimura · Y. Isoda
Service Innovation Department, NTT DOCOMO INC., Chiyoda City, Japan
e-mail: yuiko.tsunomori.fc@nttdocomo.com

T. Yoshimura
e-mail: yoshimurat@nttdocomo.com

Y. Isoda
e-mail: isoday@nttdocomo.com

R. Higashinaka
NTT Media Intelligence Laboratories, NTT Corporation, Chiyoda City, Japan
e-mail: higashinaka.ryuichiro@lab.ntt.co.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_19

have been investigated [3, 6, 9, 19, 20]. However, verifying the effectiveness of remembering and using user information in chat-oriented dialogue systems has rarely been conducted.

In this paper, we first construct a chat-oriented dialogue system that remembers and uses user information over multiple days. Then, we prepare two other systems for comparison. One only remembers user information within a single dialogue session and the other does not remember any user information. We conduct an experiment in which participants interact with the three dialogue systems for five consecutive days to evaluate the effectiveness of remembering over multiple days. As a result, we found that the effectiveness of remembering increases as time passes and that the effect of remembering can last for more than two days. We also found that remembering over multiple days, i.e., not remembering for the time being, is important to make users feel familiar with a dialogue system.

2 System Development

We first describe the user-information-extraction method. Then, we describe our proposed dialogue system that uses the acquired user information for generating system utterances.

2.1 User-Information-Extraction Method

We adopt the user-information-extraction method proposed by Hirano et al. [6], in which user information is represented in the form <[predicate-argument structures (PASs)], focus, person attribute, topic>. For example, from a user utterance “I went to England.”, we can extract the user information <[nominative=I, predicate=go, dative=England], England, experience, travel>.

To extract user information, we first determine whether user information is contained in the input user utterance. This is done by using the estimated dialogue act (DA) of an input user utterance (see Meguro et al. [11] for our DA types). When the DA is about self-disclosure, it is determined that user information is included. In this case, the following steps (1 – 4) are performed to extract user information:

1. We conduct PAS analysis [7] to extract PASs from a user utterance. PAS analysis is similar to semantic role labeling (SRL) [13] and is a technique to detect predicates together with their arguments. For example, we acquire a PAS [nominative=I, predicate=live, dative=Kyoto] from “I live in Kyoto.”
2. We extract NPs that represent the topic of the utterance, which we call the topic word or *focus* of an utterance. We use the conditional random field (CRF)-based method [10] for extraction, as described by Higashinaka et al. [5]. If the input user utterance does not contain the focus, the focus is *null*.

3. We estimate the person attribute, which represents aspects of users, included in a user utterance, as described by Hirano et al. [6]. There are 34 person attributes, e.g., “hobby”, “experience”, “ability”, “gender”, and “job”. These attributes are based on our previous analysis of personal questions [16].
4. We estimate the topic included in a user utterance, as described by Hirano et al. [6]. There are 43 topics, e.g., “travel”, “movie”, “television”, “fashion”, and “shopping”.

When the user information is obtained from a user utterance, it is stored as a record in the user-information database with a time stamp.

2.2 Development of Chat-Oriented Dialogue System That Uses Stored User Information

Our proposed chat-oriented dialogue system has two phases: greeting and chatting.

In the greeting phase, the system first introduces itself by saying its name and asking the user’s name and place of residence. We include this phase because greeting is important in social interaction.

After the greeting phase, the system enters the chatting phase (see Fig. 1 for a flowchart of the chatting phase). This phase involves the following procedure. First, the user-information-extraction method is used to extract and store user information if it is contained in the input user utterance. Then, a system utterance is generated using the following three utterance-generation units, one of which is randomly selected for utterance generation:

User-information question unit This unit asks a question to elicit user information. It does this by randomly selecting from about 200 user-information-soliciting questions prepared manually in advance. Table 1 shows some example utterances. We created the utterances referencing the questions in the Person-Database [16]. The Person-Database consists of a number of question-answer pairs created by many questioners, which cover most of the questions related to the information about users.

Utterance-generation unit based on focus This unit consists of about 20,000 hand-crafted rules (pattern-response pairs) written in artificial intelligence markup language (AIML) [19] and retrieves a response whose associated patterns match the focus of the input user utterance. The focus of the input utterance is obtained in the same manner as the user-information-extraction method. Table 2 shows an example pattern-response pair.

Utterance-generation unit based on user information This unit generates an utterance using stored user information. The time range of user information to be used is determined in accordance with the value of the parameter representing time t. If we set $t = all$, we can use the user’s information acquired in all previous dialogue sessions, and if we set $t = session$, we can use the user information

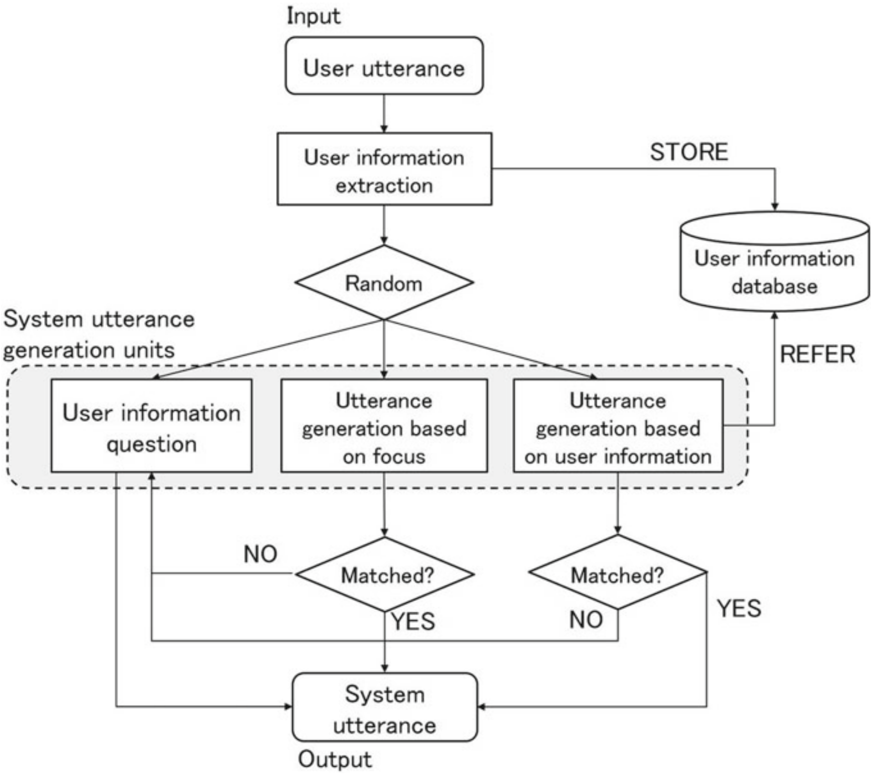


Fig. 1 Flowchart of proposed system (chatting phase). When selecting system utterance generation unit, constraint is set so that same unit cannot be used for two consecutive turns except for user-information question unit. Regarding the user-information question unit, if there exists user-information in user information database, the result of "Matched?" is YES; otherwise NO. As for the utterance generation based on user information unit, if the focus of the input user utterance matches the patterns whose associated pattern-response pairs exist in our hand-crafted rules, the result of "Matched?" is YES; otherwise NO

Table 1 Example utterances

Utterance
好きな映画はなんですか？ (What movies do you like?)
趣味は何ですか？ (What are your hobbies?)
休日は何をしてたんですか？ (What did you do in the weekend?)
学生のころは何をしていましたか？ (What did you do when you were a student?)

acquired only from the current dialogue session. If we set $t = none$, the system cannot use any user information.

The user information to be used is randomly selected from the stored user-information in user-information database (note that only the user information

Table 2 Example pattern-response pair in AIML

```
<category>
<pattern> 料理 (Food)</pattern>
<template>
料理のさしすせそは味付けの順番だけど、酒類は砂糖より前に入れるんだ。
(Regarding the order of seasonings in cooking, alcohol is put before sugar.)
</template>
</category>
```

Table 3 Example utterances prepared for each topic

Topic	Utterance
Travel	気分転換に次の週末は旅行なんてどうですか？ (How about taking a trip next weekend for a change?) 旅行を計画しませんか？ (How about planning a trip?) 友達を誘って、出かけてみたらいかがですか。(Let's take a trip with your friends.)
Movie	一緒に鑑賞しましょうよ。 (Let's go to the movies together.) 気分転換に、映画を観るのはどうですか。(How about watching a movie for a change?) ＤＶＤを観るのはどうでしょう。(How about watching a DVD?)

whose time stamp satisfies t is retrieved), and an utterance is generated using the template "By the way, Mr/Ms. [user name], didn't you talk about [focus]?". The [focus] part is filled with the focus in the selected user information; in cases in which the focus is *null*, one of the arguments of a PAS is used instead. Depending on the time between when the user information was acquired and that information is used, "a short while ago" or "previously" is added. Specifically, when using the user information acquired from the current dialogue session, "just a short while ago" is used, and when using the user information acquired before the current session, "previously" is used. In addition, we concatenate an utterance related to the topic of the retrieved user information. The utterance is randomly selected from a list of utterances prepared for each topic. Table 3 shows the example list of utterances, and there are approximately 300 utterances in all. We prepared about seven utterances for each topic by making use of typical phrases related to each topic. By having this additional utterance, the system can show interest in the topic mentioned previously by the user.

Table 4 shows example utterances from the utterance-generation units.

Table 4 Example utterances from utterance-generation units. Utterances were originally in Japanese. English translations were done by authors

Utterance-generation unit	Utterance
User-information question unit	What foods do you cook?
Utterance generation based on focus unit ($focus = food$)	Regarding the order of seasonings in cooking, alcohol is put before sugar.
Utterance generation based on user-information unit ($t = session$)	By the way, didn't you talk about hamburgers, **just a short while ago**? Why don't you cook one for a change?
Utterance generation based on user-information unit ($t = all$)	By the way, didn't you talk about cooking class, **previously**? Let's increase your repertoire of dishes.

3 Experiment

We conducted a subjective evaluation involving human participants to verify the effectiveness of our proposed dialogue system.

3.1 Experimental Procedure

Twenty-seven participants (13 males and 14 females) evaluated several dialogue systems for comparison with the following procedure.

1. Each participant followed (a) and (b) once a day for five consecutive days.
   a. The participants conducted dialogues in a text-chat interface with the three systems (see the next subsection). The participants were instructed to conduct a dialogue with each system lasting at least 30 turns; that is, more than 30 utterances from both the participant and system. The participants could end the dialogue session at any time after 30 turns.
   b. After each dialogue session, the participants evaluated the systems by responding with their degree of agreement from the statements listed in Table 5. The questionnaire items were on a seven-point Likert scale where 7 indicates the highest agreement.

   The order of the systems was randomized for each day. However, once the interaction began, the participants became aware to which system they were talking, because each system mentioned its name at the beginning of the dialogue (in this experiment, the systems were named no. 1, no. 2, and no. 3).
2. At the end of the experiment (after the last session on the fifth day), each participant ranked the three systems by considering his/her degree of user satisfaction over the entire experimental period.

Table 5 Questionnaire items

Item	Statement
Q1. Familiarity	This dialogue system sounds familiar.
Q2. Understandability	The utterance of this dialogue system is easy to understand.
Q3. Content richness	The utterance of this dialogue system is interesting and informative.
Q4. Remembering	This dialogue system remembers information about me.
Q5. Becoming familiar	I became familiar with this dialogue system.
Q6. Want to talk again	I want to talk to this dialogue system again.
Q7. Satisfaction	I am satisfied with this dialogue.

3.2 Systems for Comparison

Each system had a different t, which determines the time range of user information to be used.

a. ALL (proposed) We set $t = all$. This system uses user information acquired from all previous dialogue sessions (including the current one) to generate system utterances.
b. SESSION We set $t = session$. This system uses user information acquired only within the current dialogue session to generate system utterances.
c. NONE We set $t = none$. This system does not use any user information acquired from the dialogue to generate system utterances. For utterance generation, the system only utilizes user-information question unit and utterance-generation unit based on focus.

An example dialogue of ALL is shown in Table 6. User information <[nominative =I, predicate=have, accusative=dog], pet, property, animal> was acquired on the second day and used on the fourth day.

3.3 Results

A two-way within-subjects analysis of variance (ANOVA) was conducted to examine the effect of the systems (a–c) and the number of days (Day 1–Day 5). There was a statistically significant interaction between the systems and days for Q4 (remembering) ($F_{8,26} = 3.44$, $p < 0.05$) and Q5 (becoming familiar) ($F_{8,26} = 2.04$, $p < 0.05$); we did not find significant interaction except for Q4 and Q5. For these questionnaire items, a main effects analysis showed that the systems significantly differed from Days 2 to 5. A multiple comparison test (Holm's method) revealed that, for Q4, ALL

Table 6 Example dialogue from proposed system (ALL). User name indicates user's name acquired from system at greeting phase on first day. S denotes system utterance and U user utterance. Utterances were originally in Japanese. English translations were done by authors

		Dialogue excerpt	Utterance-generation unit
Day 2	S:	By the way, Mr. [user name], do you have any pets?	User-information question unit
	U:	I used to have a dog.	
Day 4	S:	By the way, Mr. [username], are you busy at work?	User-information question unit
	U:	I am busy, I did not get enough sleep.	
	S:	Although people talk about sleep deprivation, necessary sleeping time differs from person to person.	Utterance-generation unit based on focus
	U:	Recently, there are many apps that can help you to sleep.	
	S:	Oh, Really! By the way, didn't you talk about pets, previously? Mr. [user name], when you do not want to talk to anyone, you can relax by stroking your pet.	Utterance-generation unit based on user information ($t = all$)

was significantly better than SESSION and NONE from Days 2 to 4 ($p < 0.05$). On Day 5, ALL was significantly better than SESSION and NONE ($p < 0.05$), and SESSION was significantly better than NONE ($p < 0.05$). For Q5, from Days 2 to 5, ALL was significantly better than SESSION and NONE ($p < 0.05$), and on Day 4, ALL was significantly better than SESSION ($p < 0.05$).

From Fig. 2, we can visually confirm the better performance of ALL over SESSION and NONE for Q4 and Q5. This suggests that using user information acquired over multiple days is effective. It is also notable that since ALL outperformed SESSION, remembering over multiple days, not remembering for the time being, is effective. As the duration of the significant differences suggests, the effectiveness of the systems remain as time passed; the effectiveness of remembering took effect from Day 2 and lasted until the end of the experiment, i.e., Day 5.

In addition, to investigate the difference in overall performance among the systems, we compared the averaged subjective evaluation results over the entire experimental period (See Table 7). For each questionnaire item, we merged all ratings from all days and compared the averages. Steel-Dwass multiple comparison test showed that ALL was significantly better than SESSION in all questionnaire items except for Q3 and was significantly better than NONE in Q1, Q4, Q5, and Q6. This also indicates the effectiveness of the proposed system. It is encouraging that the ratings for Q4 and Q5 were higher than 4, which is the middle point in our 7-point Likert scale.

The average ranking that the participants ranked at the end of experiment for each system was 1.30 for ALL and 2.78 for both SESSION and NONE, showing

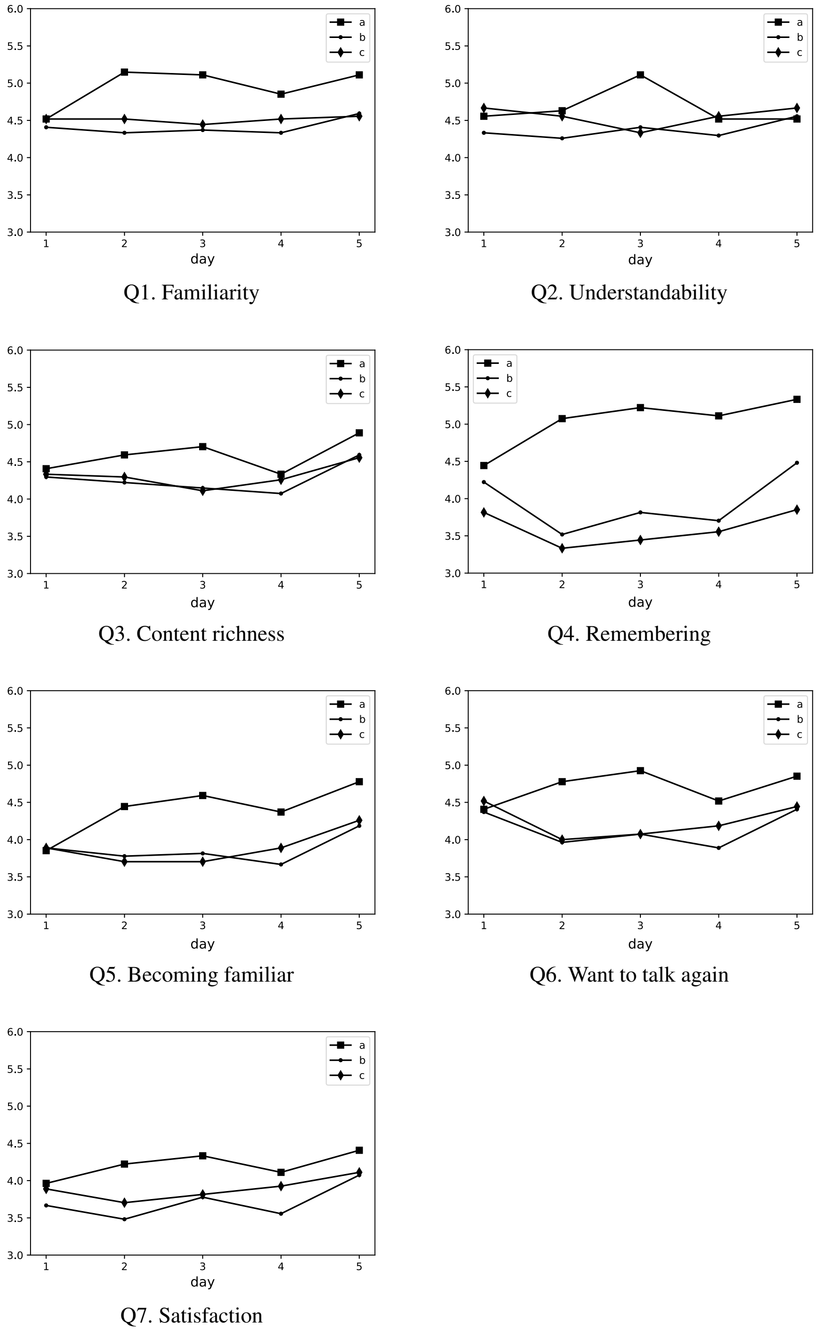


Fig. 2 Trends in subjective evaluation results for three systems (a: ALL, b: SESSION, c: NONE) over five days

Table 7 Subjective evaluation results (7 is highest) for all days. We used Steel-Dwass multiple comparison test [4] as statistical test. Superscripts a–c next to numbers indicate systems with which that value was statistically better. Double-letters (e.g., aa) mean $p < 0.01$; otherwise, $p < 0.05$

System	Q1. Familiarity	Q2. Understandability	Q3. Content richness	Q4. Remembering	Q5. Becoming familiar	Q6. Want to talk again	Q7. Satisfaction
a. ALL	4.95^{bbcc}	4.67^{b}	4.59	5.04^{bbcc}	4.41^{bbcc}	4.70^{bbc}	4.21^{bb}
b. SESSION	4.41	4.37	4.27	3.95	3.87	4.14	3.71
c. NONE	4.51	4.56	4.31	3.60	3.89	4.24	3.89

the participants' overwhelming favor for ALL. A sign test, which compared the number of times ALL won over the others, indicates that ALL is significantly better ($p < 0.01$).

4 Related Work

In task-oriented dialogue systems, research on long-term use has been conducted especially in the context of education [8] and health care [2]. Our study differs in that we focused on long-term evaluation of chat-oriented dialogue systems.

Although there has been little research on conducting evaluation of chat-oriented dialogue systems that remember and use user information, Sugo et al. [17] conducted an evaluation of their chat-oriented dialogue system that uses user information acquired from previous interaction. However, their system simply uses nouns contained in previous user utterances as user information, and they conducted only a two-session experiment; we consider it necessary to conduct a longer experiment because the effect of using user information may wear out quickly due to the decrease in the novelty of such systems. Our experiment over five days verified that the effect of using user information can last for the entire interaction period. In addition, they did not compare their system with one that only remembers user information in the current dialogue session. We believe this comparison is important so that the difference between short-term memory (remembering just for the time being) and long-term memory (remembering over multiple days) can be clarified.

5 Summary and Future Work

We proposed a chat-oriented dialogue system that uses user information acquired from dialogues and conducted a subjective evaluation of the system over five consecutive days. We found that, when a system can remember and use user information over multiple days, it improves familiarity and its effect can last during interactions. Our contribution is the following:

- As far as we know, this work is the first to verify the effectiveness of remembering on long-term interaction with a fully automated chat-oriented dialogue system.
- We verified that the effect of remembering does not diminish as time passes but can last much longer than two dialogue sessions, keeping the familiarity of the system high.

In the future, we will investigate other methods of generating system utterances using user information. We also plan to analyze the timing of effectively using user information in dialogues since we only had a random choice among utterance-generation units, which is obviously not the optimal solution. We also want to conduct a longer-term experiment because the performance of our system may decrease because of its limited capability.

References

1. Banchs RE, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In: Proceedings of the ACL 2012 system demonstrations, pp 37–42. Association for Computational Linguistics, Stroudsburg (2012)
2. Bickmore TW, Picard RW (2005) Establishing and maintaining long-term human-computer relationships. ACM Trans Comput Human Interact (TOCHI) 12(2):293–327
3. Corbin C, Morbini F, Traum D (2015) Creating a virtual neighbor. In: Natural language dialog systems and intelligent assistants, pp. 203–208. Springer, Berlin
4. Dwass M (1960) Some k-sample rank-order tests. Contributions to probability and statistics, pp. 198–202
5. Higashinaka R, Imamura K, Meguro T, Miyazaki C, Kobayashi N, Sugiyama H, Hirano T, Makino T, Matsuo Y (2014) Towards an open-domain conversational system fully based on natural language processing. In: Proceedings of the 25th international conference on computational linguistics, pp 928–939
6. Hirano T, Kobayashi N, Higashinaka R, Makino T, Matsuo Y (2015) User information extraction for personalized dialogue systems. In: Proceedings of workshop on the semantics and pragmatics of dialogue, pp 67–76 (2015)
7. Imamura K, Higashinaka R, Izumi T (2014) Predicate-argument structure analysis with zero-anaphora resolution for dialogue systems. In: Proceedings of the 25th international conference on computational linguistics, pp 806–815
8. Kasap Z, Magnenat-Thalmann N (2012) Building long-term relationships with virtual and robotic characters: the role of remembering. Vis Comput 28(1):87–97
9. Kim Y, Bang J, Choi J, Ryu S, Koo S, Lee GG (2014) Acquisition and use of long-term memory for personalized dialog systems. In: Proceedings of multimodal analyses enabling artificial agents in human-machine interaction, pp 78–87
10. Lafferty J, McCallum A, Pereira FC (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th international conference on machine learning, pp 282–289 (2001)
11. Meguro T, Minami Y, Higashinaka R, Dohsaka K (2013) Learning to control listening-oriented dialogue using partially observable Markov decision processes. ACM Trans Speech Lang Process (TSLP) 10(4):15
12. Onishi K, Yoshimura T (2014) Casual conversation technology achieving natural dialog with computers. NTT DOCOMO Techn J 15(4):16–21
13. Palmer M, Gildea D, Xue N (2010) Semantic role labeling. Synth Lect Hum Lang Technol 3(1):1–103

14. Richards D, Bransky K (2014) ForgetMeNot: What and how users expect intelligent virtual agents to recall and forget personal conversational content. Int J Hum-Comput Stud 72(5):460–476
15. Ritter A, Cherry C, Dolan WB (2011) Data-driven response generation in social media. In: Proceedings of the conference on empirical methods in natural language processing, pp 583–593. Association for Computational Linguistics, Stroudsburg (2011)
16. Sugiyama H, Meguro T, Higashinaka R, Minami Y (2014) Large-scale collection and analysis of personal question-answer pairs for conversational agents. In: Proceedings of international conference on intelligent virtual agents, pp 420–433
17. Sugo K, Hagiwara M (2014) A dialogue system with knowledge acquisition ability from user's utterance. J Jpn Soc Kansei Eng 13(4):519–526 (In Japanese)
18. Vinyals O, Le Q (2015) A neural conversational model. arXiv:1506.05869
19. Wallace RS (2009) The anatomy of A.L.I.C.E. In: Parsing the turing test, pp 181–210. Springer, Berlin (2009)
20. Weizenbaum J (1966) ELIZA-a computer program for the study of natural language communication between man and machine. Commun ACM 9(1):36–45
21. Yu Z, Xu Z, Black AW, Rudnicky AI (2016) Strategy and policy learning for non-task-oriented conversational systems. In: Proceedings of the 17th annual SIGdial meeting on discourse and dialogue, pp 404–412

Reranking of Responses Using Transfer Learning for a Retrieval-Based Chatbot

Ibrahim Taha Aksu, Nancy F. Chen, Luis Fernando D'Haro, and Rafael E. Banchs

Abstract This paper presents how to improve retrieval-based open-domain dialogue systems by re-ranking retrieved responses. The paper uses a retrieval based open domain dialogue system implemented previously, namely Iris chatbot as a case study. We investigate two approaches to re-rank the retrieved responses. The first approach trains a re-ranker using machine generated responses that were annotated by human participants through WOCHAT (Workshops and Session Series on Chatbots and Conversational Agents) and its shared-tasks [5, 6]. The second approach uses transfer learning by training the re-ranker on a large dataset from a different domain. We chose the Ubuntu dialogue dataset as the domain. The human evaluation test asked subjects to rank and review three different dialogue systems, the baseline Iris system, the Iris system enhanced with a re-ranker trained on WOCHAT data, and the Iris system enhanced with a re-ranker trained on the Ubuntu data. The Iris system enhanced with a re-ranker trained on WOCHAT data received the highest ratings from the human subjects.

I. T. Aksu
Computing 1, 13 Computing Drive, National University of Singapore,
Singapore 117417, Singapore
e-mail: taksu@u.nus.edu

N. F. Chen
Institute For Infocomm Research, A* STAR, 1 Fusionopolis Way, #21-01 Connexis, Singapore 138632, Singapore
e-mail: nfychen@i2r.a-star.edu.sg

L. F. D'Haro (✉)
Grupo de Tecnología del Habla (GTH-IPTC), ETSI de Telecomunicación, Universidad Politécnica de Madrid, Avenida Complutense 30, 28040 Madrid, Spain
e-mail: luisfernando.dharo@upm.es

R. E. Banchs
Nanyang Technological University, 50 Nanyang Ave, Singapore 639798, Singapore
e-mail: rbanchs@ntu.edu.sg

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_20

1 Introduction

Dialogue systems are often classified into two[1] categories with respect to their objectives: Task oriented dialogue systems and open domain dialogue systems [1]. Task oriented dialogue systems are designed to handle specific scenarios such as flight booking or restaurant reservations, whereas open domain dialogue systems do not focus on specific tasks to reach a target, but mostly focus on the continuity of the dialogue. In general, two approaches are used to provide responses for dialogue systems: Retrieval based models [2] and generative models [3]. The retrieval based model uses a heuristic to choose a response from a given dataset of predefined responses, whereas the generative model generates new responses. In this work, we focus on re-ranking responses for a retrieval based model.

The Iris chatbot system [4] has access to a dataset of dialogues extracted from movie scripts. At each turn it is given the user utterance, the previous utterance and the dialogue history so far. Using TF-IDF measure it finds the best matches from the dataset to the utterance and to the given dialogue history and retrieves a list of candidate answers from where the system can take one as following utterance in the dialogue.

The retrieved utterances are the best candidates to give as response but usually choosing the best one is not what a heuristic statistic can do. This is where the need for a re-ranker arises. The re-ranker is a network trained on a dataset that sorts the given candidate list with respect to their relevance to the given utterance and history, and chooses the best response to give. This paper uses two different re-rankers trained on two different datasets. One on Ubuntu dialogue corpus and the other is on a dataset that consists of annotated turns of the IRIS chatbot.

2 Related Work

Re-ranking has been commonly used in NLP problems such as parsing and translation [8], and many other studies also use it for response selection [9, 10].

Wang [11] trained two re-rankers using LSTMs. One of the re-rankers is called "strength-based re-ranker", which takes into account how often the answer to the question is encountered in related passages. The other re-ranker is called "coverage-based re-ranker", which ranks candidates higher when the union of all its contexts in different sentences could cover more aspects appearing in the question. The proposed re-rankers in this paper, different to our goal, are intended to find specific responses to a given open-domain question and therefore answers are unique. However, for a chitchat task, where there is no need to have a specific answer but to provide meaningful ones and to keep the conversation similar to what will happen when talking to a real person, the task of re-ranking answers could be more difficult as

[1] http://workshop.colips.org/wochat/.

many candidates can be selected and the selection of one before other could be due to personal or subjective reasons.

Aktolga [9] introduced a two steps approach for ranking answers to a question. They first determine the type of the answer the question will be given and then only those candidates with the correct type are compared with the question in terms of their parse structures. This ensures that answers are not accidentally ranked highly if they contain some common sentences with the question. Similar to what we mentioned above, different to a QA system where correctly determining the type of a question is important (e.g. who, when, what, etc.), for a conversational agent it could be possible, at some times, to provide a general answer or even to change the topic. For instance, it might be possible to reply to the user asking the chatbot for "Who is your favorite researcher in the area of NLP" with a "I do not know" or "this question gives me goosebumps".

Romeo [10] aims to rank the passages, retrieved as candidate answers to a question. The approach they use is to train an LSTM based network to rank similarities between the asked question and questions from the dataset. In order to do that they have two macro trees representing the original question and the candidate question which they merge to be syntactic trees of sentences composing both questions. They additionally link the trees by connecting the phrases whenever there is at least a lexical match. Our system in order to match similar questions uses word embedding which maps sequences of sentences to vectors in a an n-dimensional space. A possible future improvement to the system might be implementing the used LSTM approach in order to find similarities between questions.

All approaches mentioned above are applied to the problem of question answering (QA) systems, whereas in our work we apply response re-ranking to an open domain dialogue system. The problem we are working on has additional challenges. In QA answering systems the context is usually known which decreases the candidate answer space considerably a lot although for an open domain dialogue systems it may be not the case since the subject of the conversation can move in any direction.

3 Method

The aim of this paper is mainly to re-rank retrieved dialogues of an open domain dialogue system that is already implemented. The dialogue system has a very basic design where it has access to a dataset constituted by extracting dialogues from movie scripts. Given an utterance and the current history of the dialogue, the dialogue system finds the best match for both history and utterance and retrieves a list of responses that might be returned by using TF-IDF statistics. Two approaches are investigated.

The first approach is transfer learning, where the re-ranker is first trained on a larger out-of-domain dataset before it is applied to the target dataset. Ubuntu dialogue corpus, which is mostly on technical dialogue turns between users of the system, is used to train the re-ranker. The corpus has almost 1 million multi-turn dialogues with over 7 million utterances, it provides at each sample the history of the dialogue, the last utterance said by the user, the actual response given to that utterance and 100 randomly chosen responses for each utterance [12]. Randomly chosen responses are provided in order to give user possibly a bad or not correct example along with the correct one for each utterance. The dataset was quite divergent from the purpose of our dialogue system which is just daily talk and chit-chat. The re-ranker worked with 0.7 accuracy on the test data.

The second approach is to use a dataset with the same context of daily talk in order to train an automatic annotator that will order the list of retrieved responses. The dataset consists of annotated responses from user dialogues of the Iris chatbot [13]. Each data point has a user utterance and a chatbot response which are evaluated by annotators as valid, acceptable or invalid. The challenge in using this dataset is the limited labeled data it has. There are only 1200 turns of annotated data that can be used to train a re-ranker.

3.1 Word Embeddings

The word embeddings for both approaches were unsupervisely trained using the FastText library[2] on the Ubuntu Dialogue Corpus since it includes near 100 million words. The sentence embeddings were created by averaging embeddings of the words in the sentence.

3.2 Ubuntu Re-Ranker

The Ubuntu dataset containing almost 1 million multi-turn dialogues, with a total of over 7 million utterances and 100 million words [14], mostly consists of technical questions and replies. Since we have an open domain dialogue system in hand, using this dataset to re-rank the retrieved responses is not expected to do as good as a dataset, whose context is similar to ours. The advantage in using the specified dataset comes from its size though. This paper uses this dataset in the context of transfer learning where the idea is to store a knowledge in solving one problem and use it to solve another but related problem [15]. Data forums like Ubuntu are more widely used on Q/A systems and this is what makes our study more challenging as we try to apply it to an open domain dialogue system.

Our training data is prepared in a way to include the utterance, the actual response and one of the wrong responses at each sample. Figure 1 shows what the re-ranker network is fed and what it outputs in return. The column of random and correct response in the data is chosen randomly with a label indicating which column belongs to the correct answer. This data is fed to the network hoping that given an utterance, a random answer and a correct answer it would learn to classify which answer is the

[2]https://fasttext.cc/.

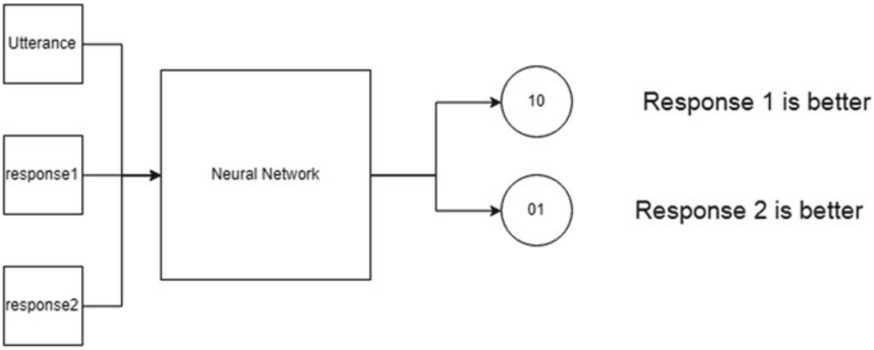


Fig. 1 Ubuntu network

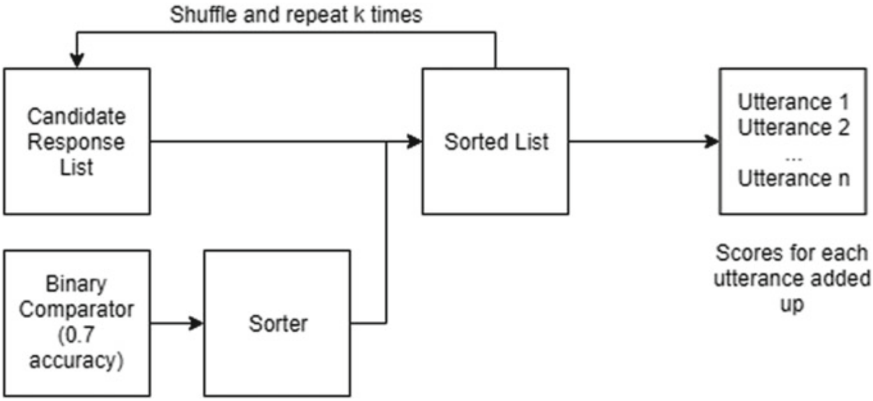


Fig. 2 Ubuntu re-ranking algorithm

one that is closer to an actual response. The network used is a multilayer perceptron with two hidden layers with 500 neurons. The learning rate is 0.001, batch size is 64. We used Adam optimizer, and had a dropout rate of 0.5. The network trained has 72% accuracy on the test data.

The next step was to write an algorithm that chooses the best response out of the candidate list given a re-ranker that provides an accuracy of 72%. The re-ranker is deterministic, so it gives the same answer every time it is fed with the same input, but the probability that the answer is correct is only 72%. The algorithm implemented is shown in Fig. 2 and does the following: First it shuffles the candidate list randomly, then it sorts it using a standard sorting function, having the re-ranker network as the binary comparator and then assigns scores to candidates according to their places in the list. This procedure is repeated several times and scores are added to candidates. At the end, the candidate with the highest score is chosen as the response. This approach works as the following: due to initial random shuffling, each call to the sorting algorithm generates unique sorting network and since 72% of the comparisons are right then as the number of turns of shuffle-order increase the sorting will be satisfactory. Since the binary comparator is not perfect, it could have some inconsistencies in the results it gives. For example lets say we have an utterance and 3 different responses A, B and C. We feed response pairs (A, B), (A, C), and (B, C) and the results the comparator gives in terms of validity is as follows respectively: $A > B$, $C > A$, $B > C$ that cannot be correct; however, our algorithm shuffles the response pairs multiple times and changes the set of binary comparisons at each turn. Thus, at the end, it is expected that it will choose the best or one of the top responses out of the n-best list.

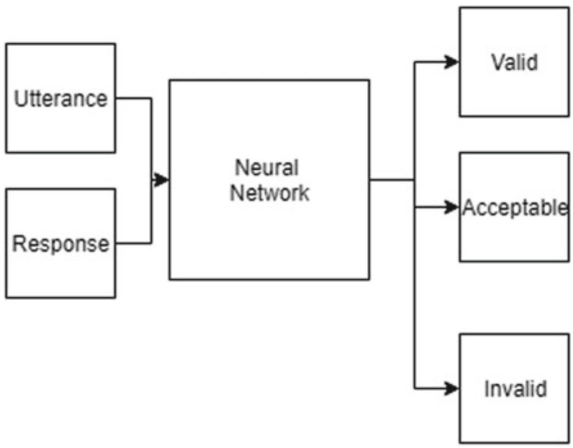
3.3 Wochat Re-Ranker

The Wochat re-ranker is trained on a dataset that consists of human annotations done on turns of IRIS chatbot's dialogues. As this dataset consists of turns created by the baseline dialogue system itself, it is a more convenient source for the training; unfortunately, its main drawback is that it has only 1500 annotated turns.

The network in this re-ranker is a multilayer perceptron with one hidden layer with 128 neurons. The learning rate is 0.008. We used Adam optimizer, and had a dropout rate of 0.8. The network, as shown in Fig. 3, is fed with an utterance and a response and classified as one of the 3 classes: Valid, acceptable or invalid. 1400 of the sample data used as training data while 100 for test and the results showed an accuracy of 75%; however, it would have been better to use k-fold cross validation to train and test the network which we did not to use during the implementation phase of the project.

The Wochat re-ranking algorithm is depicted in Fig. 4 and works as follows: the network scores each utterance according to each classification type (valid, invalid

Fig. 3 Wochat network



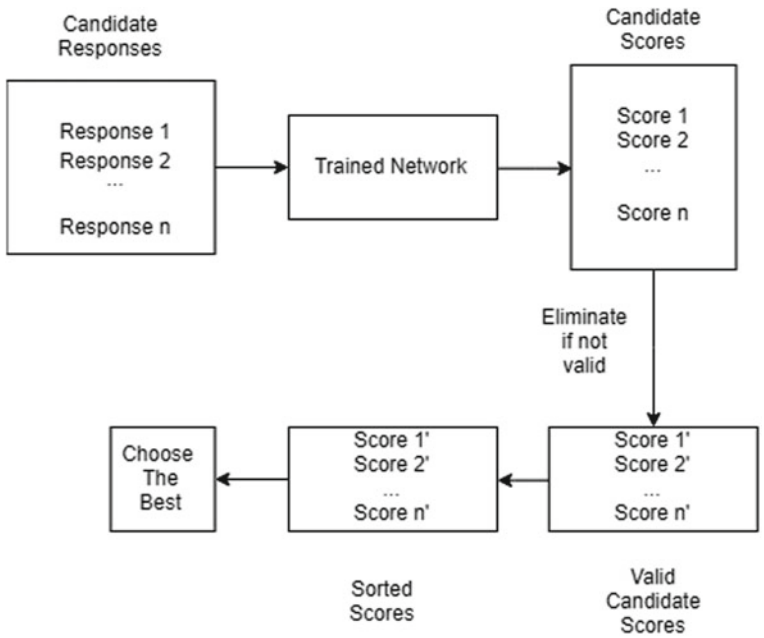


Fig. 4 Wochat re-ranking algorithm

and acceptable). Using these scores a total score for each response is found taking valid and acceptable scores as positive effects and invalid score as a negative effect on the final score. At the end, the response with the maximum score is returned. For example, an utterance might have a valid score of 0.6 an invalid score of 0.2 and an acceptable score of 0.8 while another one has 0.4, 0.7 and 0.5 respectively. In such a case, the score for the first one will be greater than the second since valid and acceptable scores are positive effects while the invalid score is a negative effect.

4 Evaluation

The evaluation of the paper was done in two ways. First by getting into similar conversations with the bots and checking their responses (Table 1). In many cases, the improvements to the responses for bots combined with re-ranker is considerable (Table 2).

The second evaluation metric is the actual users: An interface through a website, similar to [7], was implemented to get users interact with the chatbots. Since the main concern was to see the improvement of the responses of the baseline chatbot, all three bots are put anonymously to the website. Users interact with each of the bots separately without knowing their identities. In the end they report their comments

Table 1 Dialogue with the basic chatbot (Baseline or just IRIS)

Speaker	Utterance
User	Hi
Iris	Hey
User	How are you doing man
Iris	Okay..
User	What fruit do you like the most?
Iris	It's a vegetable.
User	Okay what is it?
Iris	Sir?
User	What is your favorite vegetable
Iris	Agua.

Table 2 Dialogue with the chatbot using a re-ranker trained with Wochat corpus (IRIS + Wochat re-ranker)

Speaker	Utterance
User	Hello, how are you?
Iris	Were you in an accident? Were you attacked? It's okay, I'm a doctor. my name is Miranda Grey...
User	I'm fine. So, you are a doctor, that's great
Iris	Is that it?
User	Yeah. you save lives
Iris	So why would you hesitate to lie to save lives?
User	I'm not talking about lives now. I'm surprised to know you are a doctor
Iris	What do you mean... what am i talking about? do you realize that you say that to me everyday of your life? i am dying. do you know... do you realize the death you are spreading around to the people who are around you

on each and order them from best to worst in terms of continuity and overall quality of the dialogue. There were 8 people taking the experiment and the results of the evaluations can be found in Fig. 5.

The comments are discussed through some example snippets of the chats that users had with the bots.

Table 3 shows an example snippet from a dialogue between the user and the baseline chatbot. The bot gives an overly dramatic response to the greeting coming from the user which is also irrelevant of the context.

The comments on Ubuntu chatbot are not promising as people rarely find it of good quality and define its responses as irrelevant and reckless. It does not only give short and non-informative answers but also gives answers that are out of context and

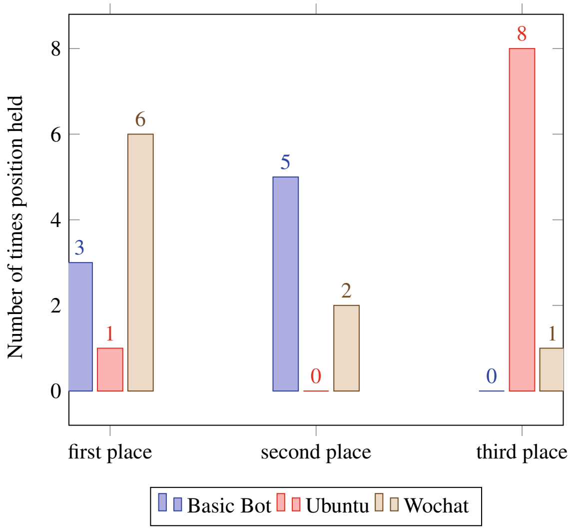


Fig. 5 Results of user evaluations, each bar refers to the number of times a bot is ranked in first, second or third place

Table 3 Dialogue with the original chatbot

Speaker	Utterance
User	Hello
Iris	Did you find it okay?
User	What do you mean?
Iris	I mean you've done a great thing. you should be rewarded monetarily
User	What did i do?
Iris	You threatened to drive her downtown. She has agoraphobia.
User	Who is she?
Iris	She's your wife!
User	But i am a woman and not gay what do you think about i?
Iris	I mean, hello!

irrelevant. Table 4 shows an example from one of the dialogues initiated with the Ubuntu chatbot.

The Wochat dialogue system seems to have a tendency to give safe and short answers which makes its answers correct but not quite informative. Users that interact with it mostly say its answers are sensible and it is just like someone who is trying to be cool. Tables 5 and 6 might be a good example that demonstrates these attributes, at most all of the answers the bot gives are acceptable, however, there are no intention of trying to get the dialogue going and providing any extra information which is not desirable.

Table 4 Example of dialogue using a re-ranker trained with Ubuntu corpus

Speaker	Utterance
User	I will need help with some school assignments
Iris	Yeah
User	It is about maths actually
Iris	weapons down
User	I got no weapons, only an old hp calculator :)
Iris	Who is it?
User	It is me Rafael
Iris	Who ?
User	Your friend from chatbot high
Iris	You hear me now?
User	Yes, let me know if you are helping with the maths assignment or not
Iris	Against
User	Ok, bye bye
Iris	Sorry about that.

Table 5 Example of dialogue using a re-ranker trained with Wochat corpus

Speaker	Utterance
User	Hello
Iris	Hello
User	How do you feel today?
Iris	Dirty
User	Why?
Iris	An opportunity
User	Are you ready for tonight?
Iris	I'm ready.
User	What do you think about the dress I showed?
Iris	I think you need help.
User	What type of help I need?
Iris	Jean?

5 Discussion

Overall with the results from Fig. 5 and all the user comments combined the Wochat bot seems to be the best one. However, there are still some improvements needed to work on it.

Regarding the problem of Wochat bot to provide short or safe answers, it can be minimized in two ways: either in the process of training the network or during the algorithm while choosing the best response. Also, checking the conversation on

Table 6 Example of dialogue using a re-ranker trained with Wochat corpus

Speaker	Utterance
User	Why are you doing this?
Iris	Doing what?
User	Saying stupid things.
Iris	It's fun
User	I don't think so.
Iris	About what?
User	Your stupid things.
Iris	It's fun
User	Who cares?
Iris	I care.
User	I think I should not talk to you anymore
Iris	Well –

Table 1, giving last few utterance matches rather than the whole dialogue history might as well improve the performance of the chatbot.

An improvement to the baseline system might be truncating the responses. Since the responses are taken directly from movie databases, they may sometimes be quite dramatic and long and the baseline gives mostly relevant even though too long answers. Thus truncating those responses might help increase the quality of the bot.

Another improvement on the systems might be filtering the responses retrieved from database before choosing one of them to give as a response. Since the Wochat dataset is composed of movie dialogues, which might turn out to be dramatic and long from time to time, some external filtering to answers such as removing short responses, duplicate responses and responses with undesirable words might increase the quality of the bot considerably [17].

Acknowledgements The authors would like to thank Esra Deniz for spending their time on long discussions on the responses of the dialogue system, Kheng Hui Yeo and Benoit Matet for their help on technical aspects and insightful discussions, and the volunteers that participated in the evaluation of the system.

References

1. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2017) A survey of available corpora for building data-driven dialogue systems. https://arxiv.org/abs/1512.05742
2. Yan R, Song Y, Wu H (2016) Learning to respond with deep neural networks for retrieval-based human-computer conversation system. http://www.ruiyan.me
3. Vinyals O, Le Q (2015) A neural conversational model. In: ARXIV (2015)

4. Banchs RE, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In: ACL 12 proceedings of the ACL 2012 system demonstrations
5. Kong-Vega N, Shen M, Mo W, D'Haro LF (2018) Subjective annotation and evaluation of three different chatbots WOCHAT: shared task report. IWSDS, Singapore
6. D'Haro LF, Shawar BA, Yu Z (2016) RE-WOCHAT 2016-shared task description report. Workshop on collecting and generating resources for chatbots and conversational agents-development and evaluation (RE-WOCHAT)
7. Lin L, D'Haro LF, Banchs R (2016) A web-based platform for collection of human-chatbot interactions. In: Proceedings of the fourth international conference on human agent interaction. ACM, New York
8. Quan VH, Federico M, Cettolo M (2005) Integrated N-best Re-ranking for spoken language translation. In: Proceedings of the 9th European conference on speech communication and technology, Interspeech-05
9. Aktolga E, Allan J, Smith DA (2011) Passage reranking for question answering using syntactic structures and answer types. Lect Notes Comput Sci 6611:617–628
10. Romeo S, Martino GDS, ón-Cedeño AB, Moschitti A, Belinkov Y, Hsu W, Zhang Y, Mohtarami M, Glass JR (2016) Neural attention for learning to rank questions in community question answering, COLING
11. Wang S, Yu M, Jiang J, Zhang W, Guo X, Chang S, Klinger ZWT, Tesauro G, Campbell M (2018) Evidence aggregation for answer re-ranking in open-domain question answering. In: International conference on computational linguistic (2018)
12. Lowe RT, Pow N, Serban IV, Charlin L, Liu CW, Pineau J (2017) Training end-to-end dialogue systems with the ubuntu dialogue corpus
13. D'Haro LF, Banchs RE (2017) Learning to predict the adequacy of answers in chat-oriented humanagent dialogs. Lecture notes in computer science, 6611
14. Lowe R, Pow N, Serban I, Pineau J (1995) The ubuntu dialogue corpus: a large dataset for research in unstructured multi-turn dialogue systems. https://arxiv.org/abs/1506.08909
15. Yu J, Qiu M, Jiang J, Huang J, Song S, Chu W, Chen H (2017) Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. https://arxiv.org/abs/1711.08726
16. Hulth A (2003) Improved automatic keyword extraction given more linguistic knowledge Proceedings of EMNLP '03, ACL, 2003
17. Yusupov I, Kuratov Y (2017) NIPS conversational intelligence challenge 2017 winner system: skill-based conversational agent with supervised dialog manager. In: International conference on computational linguistic

Online FAQ Chatbot for Customer Support

Thi Ly Vu, Kyaw Zin Tun, Chng Eng-Siong, and Rafael E. Banchs

Abstract Chatbots and conversational systems are becoming a prominent research area, and many businesses are starting to leverage on their capability to handle basic communication tasks. With a vast variety of available frameworks for chat-bot development from tech giants, business organizations can build their own systems quickly and conveniently. However, these frameworks often lack a proper set of holistic tools to build a chatbot that is manageable, adaptable to learn, and scalable. Hence, frequently, additional machine learning mechanisms are needed to improve performance. In this paper, we demonstrate a chatbot system that uses machine learning to answer Frequently Asked Questions (FAQs) from our school website. The system includes different types of user query and a vector similarity analysis component to handle long and complex user queries. In addition, the Google' s DialogFlow framework is used for intention detection.

1 Introduction

With the advancements in natural language[1] understanding (NLU) and speech recognition technology, chatbots are able to resolve some of the problems in customer services with a reasonable cost. They offer a new opportunity to reduce cost for

[1] https://dialogflow.com/.

T. L. Vu (✉) · K. Z. Tun · C. Eng-Siong · R. E. Banchs
School of Computer Science and Engineering, NTU, Nanyang Ave, Singapore
e-mail: tlvu@ntu.edu.sg

K. Z. Tun
e-mail: ztkyaw@ntu.edu.sg

C. Eng-Siong
e-mail: aseschng@ntu.edu.sg

R. E. Banchs
e-mail: rbanchs@ntu.edu.sg

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_21

companies, big or small, in almost any sector. A customer support or FAQ bot will never quit, never get bored with all the questions it receives. It can be faster than humans in learning about the company products or services, and always providing the exact and detailed information. Chatbots are also better suited than humans in handling multiple requests at a time, without being distracted or tired.

Like a Question Answering system (QA system), a chatbot is also an interactive system able to answer user questions in natural language [4]. However, chatbot applications allow for interaction in a conversational format, providing a better and improved user experience. Hence, they have become the preferred choice over conventional QA systems. While the QA systems mainly focus on identifying the question type, chatbot applications focus on a more comprehensive processing of the query for understanding the intention, extracting additional information (entities) and keeping track of the conversation context. Additionally, chatbot applications are intended to deliver a better user experience by means of character style and impersonation, making the interaction more human-like [1, 3]. Recently, the number of available frameworks for creating chatbot applications has increased significantly, from open source to enterprise level, from free to paid service,[2] from ready to use solutions to highly customized frameworks.[3]

In this work, we present a chatbot system that integrates a highly scalable and customized framework along with additional machine learning techniques to answer FAQs from our school website (NTU' s School of Computer Science and Engineering, Singapore).[4] As main platform, we selected DialogFlow. The evaluation from [2] showed that the DialogFlow performed worst as the NLU service with all the experimented corpus, but it supported integration to many messenger platforms and website, fulfillment service and other features. We also integrated additional processing components including a question generation module to generate permutations, sentence similarity analysis from spaCy.[5] The data from question generation is to diversify the training data, to handle different queries from users. The sentence similarity module strengthen the language understanding part. Our target was to build a system not only for experimental purposes, but also for deployment, requiring it to be scalable and modular for future integration of new components to improve its performance.

2 System Architecture and Implementation

Figure 1 illustrates the system architecture of the developed FAQ chatbot. In this section, we will describe in detail the different components of the system.

---

[2] https://medium.com/mindlayer/a-generic-summary-of-chatbot-public-apis-26448c1b108c.

[3] https://tryolabs.com/blog/2017/01/25/building-a-chatbot-analysis--limitations-of-modern-platforms/.

[4] http://askntu.ntu.edu.sg/home/ntu_wide/ifaq.aspx.

[5] https://spacy.io/usage/vectors-similarity.

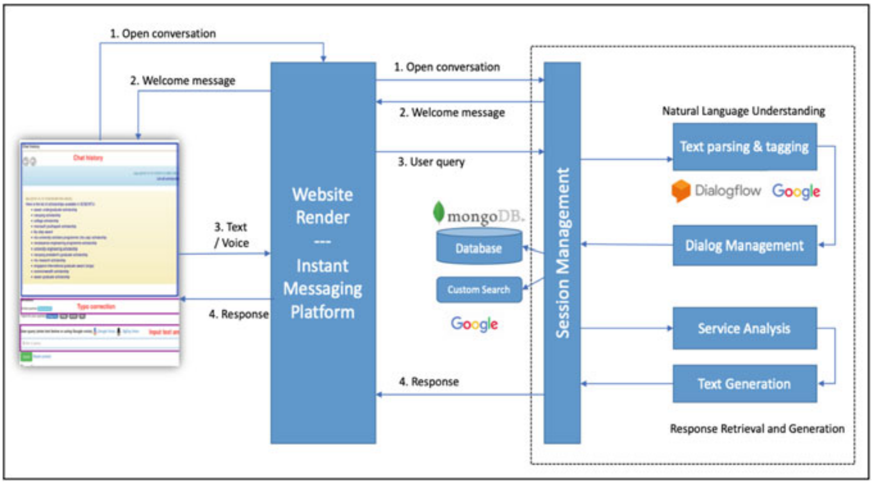


Fig. 1 NTU/SCSE FAQ chatbot system architecture

Fig. 2 Web interfaces for data management (left) and agent training (right). In the data management interface, system admin could review and modify the training example. With agent training, the list of untrained examples (question, answer, and meta information) are displayed. When system admin edits the example, he/she can select the closest intention in the fourth column. After finishes reviewing, he/she can do batch training by click to Training button at the top of the page

The main components of the system are: (1) a user interface to receive user queries and display system responses, (2) a FAQs agent to handle language understanding and dialog management, (3) a web-service to process the back-end tasks, including information retrieval from a database [5], information searching from predefined websites (Google Custom Search). We also added other supported modules, such as a crawler to collect data from the school website, agent training to adapt with new data, data management, integrated to our website (Fig. 2).

We also implemented two additional sub-components to improve the chatbot performance: a module to compute sentence similarity to select the best response, and a question generation module to provide more training data for chatbot agent.

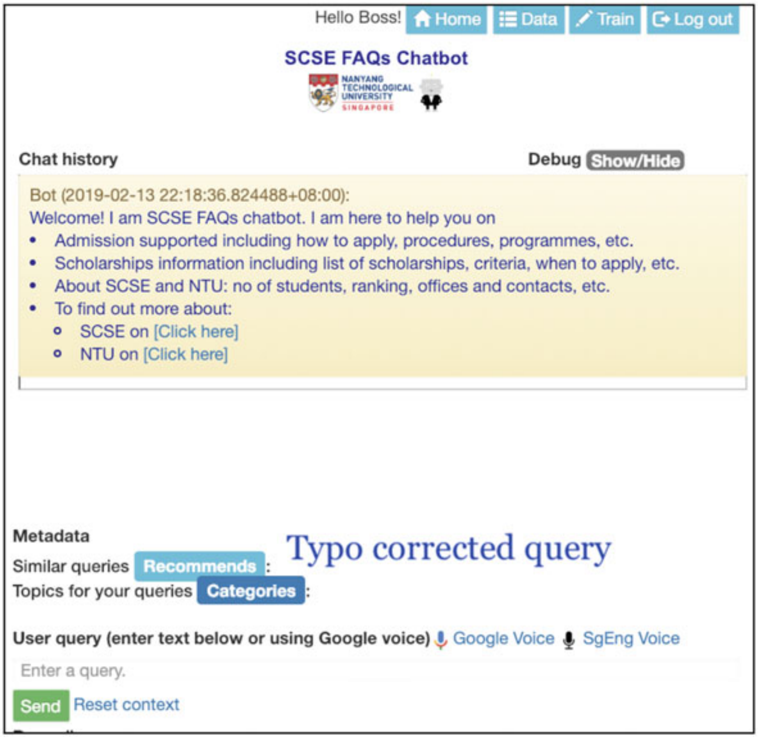


Fig. 3 User interface for conversation with user. **a** User can have a conversation with the chatbot via text or speech (We are using two speech engines: Google Voice and our in-house speech engine). The input query (or transcription from speech engine) is entered in the User query text field. When user presses Send button, the query will be sent to the chatbot agent. The back-end (Dialogflow agent and our webservice) processes the query, returns the response and displays output text in the Chat history area

2.1 Web User Interface

We use Python and Flask framework to build the website, as illustrated in Fig. 3. It has a chat history to store the conversation log between user and system, a debugging panel to analyze the correctness of intent/entities extraction, similar queries to display the suggestion of typos corrected queries.[6]

[6]https://abiword.github.io/enchant/.

2.2 Language Understanding and Dialog Manager

Our FAQs agent is created on the DialogFlow platform, and focused on text-based conversations. Since our FAQs chatbot is for the school domain, we focused on intents and entities about these four relevant topics: admission, programmes, scholarships and general school inquiries. First, we defined the list of entities: scholarships, programmes, `student_groups`, `fees_type`, and so on. Second, we created the list of intents to resolve the most relevant intentions: "How to apply to engineering school?", "Which programme can I apply?", 'Can I get more information about Nanyang Scholarship?", etc. We reviewed subset questions (655 question and answer pairs) in the data set (2600 question and answer pairs) and grouped similar questions into single intents. This was done manually at first, and improved over time. As more examples are provided, the intent recognition accuracy can be improved (Tables 1, 2 and 3).

We grouped the list of intentions into 3 main categories: the simple intention (their responses can be hard-coded in text response of the DialogFlow framework), the structured intentions (their responses have to be retrieved from the database), the complex intentions (their response is the response of the most similar query). With structured and complex intentions, we have to enable webhook option so their response will be forwarded and processed by a web-service.

Not like other dialog systems, intentions with context were not the common cases in this type of chatbot (question and answering). However, in some cases, the context make the conversation more flexible and naturalness. Here is the typical conversation where the context information is useful:

Table 1 Data used in our FAQs agent

Item	Quantity
Number of questions, answers crawled from askntu	2600
Number of questions, queries used to train the agent	655
Number of intentions	68
Average number of queries per intention	9.6

Table 2 Number of queries and intents in similar queries group in FAQs chatbot

Item	Quantity
Number of questions, queries	66
Number of intentions	7
Average number of queries per intention	9.4

Table 3 Queries test set for system evaluation

Query type	Quantity	Result	Comment
Matched queries	10	8 correct intents	The response is still long and containing other information
Permutation queries	8	6 correct intents	The response is the most similar query instead of response for its
New queries	5	0 correct intent	These queries are all sent to Google Custom Search module

- User: List all scholarships that NTU offers?
- System: Here is the list of scholarships that NTU offers:
- User: Tell me about the last one
- System: <show the general information about Nanyang Scholarship>
- User: When is the close date for this scholarship application
- System: The close-date of this scholarship application is on <date-time>

2.3 *Web-Service to Retrieve Information*

To provide more flexible and customized answers, we implemented a web-service for DialogFlow webhook calls. This service receives calls from the DialogFlow framework with information that includes the recognized intent and extracted entities. These requests are categorized into three main groups: structured queries, similar queries and custom queries.

The first group, structured queries, requires information about intent name and entities to construct database queries. While the intent refers to specific table or document names, entities are used as filter conditions. An SQL query is constructed with these filter conditions and intent name to retrieve a response from the database. The challenge for serving this kind of query is the completeness of the database information. If there is no such an answer or data, the SQL query will retrieve nothing from the database. Data in the first group was reviewed and constructed into 13 classes in MongoDB.[7]

The second group of requests is composed by queries with a high degree of complexity and which typically do not contain entities. An example of these complex queries is: "I have obtained sub-pass in the mother tongue language and i did not

[7] https://docs.mongodb.com/manual/tutorial/.

take the mother tongue syllabus B. I have done well in my A level subjects will i be given a provisional offer of admission?"

These queries are very hard to process, and typically, it is not possible to compose structured queries from these types of questions. Therefore, we come up with and alternative solution, which consist of comparing with the known set of questions to find the most similar question, and get the corresponding answer for it. The steps to build this module are two:

1. Store the list of known complex queries, grouped by intent or topic, and their answer in an index. We experimented with 7 intentions, from 66 user queries.
2. Whenever the chatbot receives a query, it will compare with the list of available questions, select the most similar and return its response. Semantic similarity is calculated using the spaCy library.

The final group, custom queries, are not handled by our FAQs agent. Queries in this group will be used to search in Google with predefined list. Top ten results will be formatted and displayed in the web-interface, user can click to the link and check further.

2.4 Sentence Similarity Module

We tried two models to measure the sentence similarities: Google Word2Vec and spaCy. We selected the model and similarity measurement from spaCy since it is balanced between the accuracy and performance of processing.

The procedures to find the response from similar queries with spaCy and Google word2vec are illustrated in Algorithms 1 and 2, respectively. Semantic similarity is measured by cosine similarity. In our experiment with Google word2vec, we selected DELTA = 0.85, weighing the semantic similarity more than the word order similarity.

Algorithm 1 Sentence Similarity algorithm with spaCy

```
procedure get_response_from_similar_query_spacy(intentname, userquery)
  max_similarity ← 0.0
  ref_sentences ← get_ref_from_intent_name(intentname)
  for ref_sentence ← ref_sentences do
    ref_doc ← nlp(ref_sentence)
    input_doc ← nlp(userquery)
    semantic_score ← doc_similarity(ref_doc, input_doc)
    if max_similarity < semantic_score then
      max_similarity = semantic_score
  return max_similarity
```

Algorithm 2 Sentence Similarity algorithm with Google word2vec

```
procedure get_response_from_similar_query(intentname, userquery)
  max_similarity ← 0.0
  ref_sentences ← get_ref_from_intent_name(intentname)
  for ref_sentence ← ref_sentences do
    ref_tokens ← split(ref_sentence)
    input_tokens ← split(userquery)
    ref_sent_vector ← get_wvector_from_sentence(ref_tokens)
    input_sent_vector ← get_wvector_from_sentence(input_tokens)
    semantic_score ← semantic_similarity(ref_sent_vector, input_sent_vector)
      word_order_score ← word_order_similarity(ref_sent_vector, input_sent_vector)
      r ← DELTA * semantic_score + (1.0 − DELTA) * word_order_score
      if max_similarity < r then
        max_similarity = r
    return max_similarity
```

2.5 Other Modules

In addition the main FAQ related capabilities, the following features have been also integrated into the system:

- Chitchat. To make the chabot more friendly and interactive with real users, we also used the small-talk feature provided by DialogFlow. This enables the chatbot to support out-of-topic conversations about itself, saying hello or goodbye or other chitchats.
- User feedback. Getting explicit feedback from users is also useful for improving and adjusting the system responses. To this end, we also develop a feedback feature, which the user can provide any time after each query is responded.
- Spelling correction. Our system also includes a sentence correction feature, which checks typos in user queries by using a standard English dictionary. A sentence suggestion module will check the input sentence on the fly. We used the python library of enchant, with addition words.
- Web crawler. To collect data used for our chatbot, we write the program, using the Selenium framework and Sukhoi[8] to browse the FAQs website of NTU to get the asked questions and answers. After processed and removed HTML tags and other meta-information, we had approximate 2600 question and answer pairs, covered many topics.

[8]https://pypi.org/project/sukhoi/.

3 Evaluation

To evaluate our chatbot, we conducted two types of experiments. The first experiment is to evaluate the accuracy of the system functions, with selected queries, cover from known scenarios to the unexpected scenarios. The queries test set is including some matched queries with the training data set, permutation queries (from question generation output), and also completely new queries. The results showed in the table below. The system handles the trained intentions quite well, but fails to capture the new queries.

4 Conclusion

Our target is to build a system that is both scalable and modular, so it is easy to add or remove other modules when necessary. To further improve the system, the following additional modules need to be implemented: (a) Collecting and analyzing the user evaluation, (b) Using output of Question Generation to improve the accuracy, (c) Text summarization to provide the flexible and short response in conversation style and (d) support SPARQL database search

Acknowledgements This research was supported by the Speech team in Multimedia and Interactive Computing Lab (MICL), School of Computer Science and Engineering, NTU, Singapore.

References

1. Abu Shawar B, Atwell E (2007) Different measurements metrics to evaluate achatbot system. In: Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies, pp. 89–96. Association for Computational Linguistics
2. Braun D, Hernandez-Mendez A, Matthes F, Langen M (2017) Evaluating natural language understanding services for conversational question answering systems. In: Proceedings of the 18th Annual SIG dial Meeting on Discourse and Dialogue
3. Kuligowska K (2015) Commercial chatbot: performance evaluation, usability metrics and quality standards of embodied conversational agents
4. Quarteroni S, Manandhar S (2007) SA chatbot-based interactive question answering system. Decalog 2007: 83
5. Yan Z, Duan N, Bao J, Chen P, Zhou M, Li Z (2016) JiansheZhou: Docchat: an information retrieval approach for chatbot engines using unstructured documents. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol 1, pp 516–525

What's Chat and Where to Find It

Emer Gilmartin

Abstract Chat or 'non-goal directed' dialogue has becomse a popular domain for spoken dialog system research, while the exponential increase in the use of commercial chatbots is creating interest in how to add friendly talk to make task-oriented systems more personable, or indeed to create systems which can create and maintain friendly relations with a user through the use of social talk. However, such talk is not very well defined, relevant data sources are few, and how to create artificial social talk is still an inexact science. This non-technical position paper briefly overviews these areas, exploring data used in chat systems and the limitations and challenges involved, and how these impact on the implementation of realistic social talk in spoken dialog systems.

1 Introduction—What's Chat?

Human spoken interaction takes many forms, from highly formalised rituals and ceremonies to light and aimless casual chat. Interaction types or speech exchange systems can be differentiated along a number of dimensions including goal, formality/register, length and frequency of contributions, number of participants, and relationship between participants. A common division for classification is between 'instrumental', transactional or task-based dialog and 'interactional', interpersonal, or social talk [4]. Task-based dialog is generally considered to cover practical tasks mediated through speech—such as commercial transactions, professional consultations, business meetings and discussions or problem-solving sessions. Social talk ranges from short interactions in public places between strangers ('bus stop talk'), through intermittent social talk between co-workers throughout the day, stretches of casual talk which bookend or punctuate transactional interactions, to long private conversations between intimate friends ('dinner party talk') [8, 20]. The content of

E. Gilmartin (✉)
ADAPT Centre, Trinity College Dublin, Dublin, Ireland
e-mail: gilmare@tcd.ie

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_22

and markers of success in task-based talk are defined by its goal—there are usually a limited number of sequences or sub-dialogues which together constitute the interaction, and reaching a defined result such as the successful sale of a product, provision of information, or decisions on agenda items, allows the dialog to be considered successful. Social talk is *not* goal-less, there is a broad consensus among researchers in a variety of fields that such talk serves strong social goals—building and maintaining social bonds, informing participants of their interlocutors' personality, values, feelings and affect [1, 6, 12, 16, 24]. However, these are not short-term goals, fulfilled in one interaction, and thus assessment of conversational success, beyond simple measurement of time users spend chatting to the system, is challenging.

Casual or social talk is thought to involve a levelling of role, power and status differentials between interlocutors, with equally distributed speaker rights [5, 15]. Topics are decided locally and proposable by any participant. This leads to unpredictability at the level of topic change, although there is internal coherence within topics. The structure of casual talk is not normally a series of questions and short answers as often found in task-based dialog, but rather a series of statements and comments on a topic, often interspersed with backchannels and short comments with little or no additional information, a phenomenon known as 'idling' [20]. In addition, longer conversations devolve into phases of casual interactive 'chat' and more monologic 'chunks' of narrative or extended opinion. It is thus not possible to model such conversations as a series of adjacency pairs [22]. This chat/chunk form also leads to challenges for endpointing, as within speaker pauses in chunk phases or long turns can be longer than between speaker pauses in interactive chat, and thus a single endpointing module will not suffice.

2 Where to Find It: Data and Generation

Human-human casual conversational data is in short supply. Most dialog datasets, and spoken dialog corpora in particular, are collections of task-based conversations, either real or staged meetings or elicited dialog created using knowledge gap tasks [2, 13, 17]. Collections of casual talk do exist [21], but are often recordings of 'first encounters' or of short conversations [3, 7, 18], and are not large enough for training of stochastic systems and end-to-end models in particular. Corpora of telephone conversations do provide longer conversations, which move into chat/chunk format after initial volleys of short question-answer pairs.

The structure and content of social talk makes any stochastic process of finding the best next utterance for a system very challenging. As most utterances in longer casual conversations are not questions, and do not have a highly predictable response, unless a conversation is closely following an example in training data and thus running the danger of overfitting, it is highly unlikely that a suitable content rich response will be proposed—systems often generate very common responses ('ok', 'yeah'). This means that the system is not outputting talk that moves the conversation along or provides something for the interlocutor to comment on.

Dialog is unpredictable, beyond simple 'getting to know you' question-answer sequences, which only occur at the beginning of any series of interactions. Several systems have used the concept of dialog acts (moves, intents, etc.) to abstract away from variable surface forms of user utterances. An act labelled 'request[depDate]' could map 'When does my flight leave?' or 'Can you tell me the departure date for my flight, please?' to the same underlying user intent and match to the best next system action [23]. This delexicalisation process is useful for predictable task-based dialogs, but once casual talk is examined it becomes clear that there is not a uniform adjacency pair structure where the first part utterance of a pair (somewhat!) reliably projects the second part, as occurs in task-based talk [19]. Indeed, the vast bulk of dialog acts in casual talk are informs, and often sequences of informs from one speaker. Even in task-based dialog, human-human interaction often involves multi-intent or multi-act utterances, where the speaker closes one adjacency pair and opens another, sometimes in a different domain—for example answering a question on a flight booking, and asking about the weather in a location.

Telephone corpora, such as Switchboard [11], may be useful for modelling longer chunk phases, and the transitions between chunks could serve as useful models of transitions comprising short stretc.hes of chat between chunks. In general, for conversations longer than simple greetings and short smalltalk, it would seem logical to view the interaction as a series of sub-conversations, and model accordingly on multiple datasets. Such 'blended' dialog is a promising avenue of research. However, although it will be possible to provide longer stretches of monologic system talk which are relevant, this may 'raise the stakes' for system understanding of the user, who will be prompted to provide longer stretches of monologue, with attendant need for backchannelling models as well as robust natural language understanding capability and context modelling spanning multiple previous turns. These challenges have made evident by work on chat system evaluation [14].

3 The ADELE Project

We are currently working on several of the challenges mentioned above, in order to gain insight into the mechanics of casual talk in order to more accurately model social dialogue for use in applications in elder care and education.

To explore the dialog acts present in social talk, we built and annotated a corpus of over 200 text dialogs where participants tried to discover commonalities in personas given to them, thus eliciting 'getting to know you' dialog [9]. Even in this relatively predictable interaction, informs and comments (declaratives) outnumbered questions and requests by a factor of 3.4—in longer conversations, this disparity is greater with increasing prevalence of chunks dominated by one speaker. We have also found that the variety of social acts present in these limited dialogs is not adequately covered by existing annotation schemes and are working on defining new acts to make the task of abstraction easier and to aid understanding of the structure of even the simplest formulaic exchange of greetings and smalltalk [10]. We have built a stochastic system for the ConvAI Challenge, based on a 'getting to know you' paradigm similar to the ADELE corpus—this experience has highlighted the difficulties of using a data-driven approach to select responses which are appropriate and more than simple 'idling' or platitudes.

We are now working on the Switchboard corpus to understand the structure of longer monologic stretches and on how transitions between chunks and chat are managed. We have annotated 200 conversations from the corpus for topic and topic shift, and are now working on modelling these transitions. We hope this work will allow us to further explore possibilities of blended dialog, and generate realistic transitions between stretches drawn from separate sources.

As with most research in this field, we are severely hampered by the lack of relevant data in adequate quantities, and would hope that greater interest in and discussion of the challenges of creating realistic social talk either as standalone systems or within task-based applications will lead to the creation of relevant datasets open to the community.

Acknowledgements This work is supported by Science Foundation Ireland (Grant 13/RC/2106) and the ADAPT Centre (www.adaptcentre.ie) at Trinity College, Dublin.

References

1. Abercrombie D (1956) Problems and principles: studies in the teaching of english as a second language. Longmans, Green
2. Anderson A, Bader M, Bard E, Boyle E, Doherty G, Garrod S, Isard S, Kowtko J, McAllister J, Miller J et al (1991) The HCRC map task corpus. Lang Speech 34(4):351–366
3. Aubrey AJ, Marshall D, Rosin PL, Vandeventer J, Cunningham DW, Wallraven C (2013) Cardiff conversation database (CCDb): a database of natural dyadic conversations. In: 2013 IEEE conference on computer vision and pattern recognition workshops (CVPRW), pp 277–282. https://doi.org/10.1109/CVPRW.2013.48
4. Brown G, Yule G (1983) Teaching the spoken language, vol 2. Cambridge University Press
5. Cheepen C (1988) The predictability of informal conversation. Pinter London
6. Dunbar R (1998) Grooming, gossip, and the evolution of language. Harvard University Press
7. Edlund J, Beskow J, Elenius K, Hellmer K, Strömbergsson S, House D (2010) Spontal: a swedish spontaneous dialogue corpus of audio, video and motion capture. In: LREC
8. Eggins S, Slade D (2004) Analysing casual conversation. Equinox Publishing Ltd
9. Gilmartin E, Spillane B, O'Reilly M, Su K, Saam C, Cowan BR, Campbell N, Wade V (2017a) Dialog acts in greeting and leave taking in social talk. In: Proceedings of the 1st ACM SIGCHI International Workshop on Investigating Social Interactions with Artificial Agents. ACM, New York, NY, USA, ISIAA 2017, pp 29–30. https://doi.org/10.1145/3139491.3139493
10. Gilmartin E, Spillane B, O'Reilly M, Su K, Saam C, Cowan BR, Campbell N, Wade V (2017b) Dialog acts in greeting and leavetaking in social talk. In: Proceedings of the 1st ACM SIGCHI international workshop on investigating social interactions with artificial agents. ACM, pp 29–30
11. Godfrey JJ, Holliman EC, McDaniel J (1992) SWITCHBOARD: Telephone speech corpus for research and development. In: 1992 IEEE international conference on acoustics, speech, and signal processing, 1992. ICASSP-9,. vol 1, pp 517–520
12. Hayakawa SI (1990) Language in thought and action. Houghton Mifflin Harcourt

13. Janin A, Baron D, Edwards J, Ellis D, Gelbart D, Morgan N, Peskin B, Pfau T, Shriberg E, Stolcke A (2003) The ICSI meeting corpus. In: 2003 IEEE international conference on acoustics, speech, and signal processing, 2003. Proceedings.(ICASSP'03), vol 1, pp I–364
14. Kong-Vega N, Mingxin S, Mo W, D'Haro LF (2020) Subjective annotation and evaluation of three different chatbots wochat: shared task report
15. Laver J (1975) Communicative functions of phatic communion. Organization of behavior in face-to-face interaction, pp 215–238
16. Malinowski B (1923) The problem of meaning in primitive languages. In: Supplementary in the meaning of meaning, pp 1–84
17. McCowan I, Carletta J, Kraaij W, Ashby S, Bourban S, Flynn M, Guillemot M, Hain T, Kadlec J, Karaiskos V (2005) The AMI meeting corpus. In: Proceedings of the 5th international conference on methods and techniques in behavioral research, vol 88
18. Paggio P, Allwood J, Ahlsén E, Jokinen K (2010) The NOMCO multimodal Nordic resource–goals and characteristics
19. Schegloff E (1988) Presequences and indirection: applying speech act theory to ordinary conversation. J Pragm 12(1):55–62
20. Schneider KP (1988) Small talk: Analysing phatic discourse, vol 1. Hitzeroth Marburg
21. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2015) A survey of available corpora for building data-driven dialogue systems. arXiv:151205742 [cs, stat]
22. Thornbury S, Slade D (2006) Conversation: From description to pedagogy. Cambridge University Press
23. Traum DR, Hinkelman EA (1992) Conversation acts in task-oriented spoken dialogue. Comput Intell 8(3):575–599
24. Turnbull W (2003) Language in action: psychological models of conversation. Routledge

Generation of Objections Using Topic and Claim Information in Debate Dialogue System

Kazuaki Furumai, Tetsuya Takiguchi, and Yasuo Ariki

Abstract In recent years, systems with a dialogue interface are attracting wide attention [1, 2]. We propose a dialogue system that can debate with users about news broadcasts on TV or radio and help users to understand the meaning deeply. We previously reported a debate system that collected opinions from the Web [4], vectorized them, and finally selected the most appropriate supporting/opposing opinion among them for debating. In this paper, we propose a Neural Network Language Model that can generate objections instead selecting one opinion for debating. The model generates sentences by putting claim information (supporting/opposition) in the input layer of Long Short-Term Memory (LSTM) [3]. We conducted experiments by BLEU score and Human Evaluation, and both showed the effectiveness of our method.

1 Introduction

There are many systems that support users by answering their questions [1, 2], but in order to deal with even more complicated problems, we propose a debate dialogue system that supports users to be able to understand things deeply by providing new perspectives on topics of news broadcast on TV or radio. To this end, we already developed a system that could estimate a user's claim (supporting/opposing) on the topic as well as the reason behind the claim, and debate with a user by showing the appropriate opinions selected from the Web [4]. However, depending on the number and quality of opinions on the Web, we encountered the problem that the debate was

K. Furumai (✉) · T. Takiguchi · Y. Ariki
Kobe University, Nada Ward, Japan
e-mail: kazuaki.furumai@stu.kobe-u.ac.jp

T. Takiguchi
e-mail: takigu@kobe-u.ac.jp

Y. Ariki
e-mail: ariki@kobe-u.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_23

not active in some cases. Therefore, in this paper, we propose a method to generate opinions or objections that are more appropriate to the user's opinion when there are no suitable candidate opinions found on the Web.

For generating objections, we employ a neural network language model using LSTM. Unlike the majority of seq2seq models, our model does not have the encoder of the input sentence but, rather, is trained to generate an objection to the user's opinion, by decoding the document vector created by Sparse Composite Document Vectors (SCDV) [6]. SCDV combines syntax and semantics learnt by word embedding models together with a latent topic model that can handle different senses of words, thus enhancing the expressive power of the document vectors. Specifically, we cluster distributed representations of all words using Gaussian Mixture Models, improve the word representation based on the probabilities belonging to each class, and use it for calculating the document vector. In addition, we control the claim of the sentences generated by the model by connecting a system claim vector (that is opposite to the user's claim) to the word embedding vector. We examined the performance of this model by BLEU score and a subjective evaluation experiment.

2 Debate Management

In this section, we briefly explain the process of objection generation in our debate dialogue system, which has already been proposed in [4], (please see the upper part of Fig. 1). First, the Language Understanding module estimates the user's claim (supporting/opposing/neither) and reason (presence/absence). We use a Convolutional Neural Networks model proposed by Shi [5] to estimate them. In this module, when the user's claim is estimated to be "neither", or when the reason is estimated to be "absence", the system generates an utterance to ask the user for clarification.

Finally, after the system estimates the user's claim and reason, the system selects an opinion from the debate database, that is against the user's claim. For example, if the user's claim is estimated to be "supporting", the system utterance is selected from the opposite opinion stored in the debate database. As for the selection method, an opposite opinion with the highest cosine similarity to the user's utterance is selected.

However, because there are cases where it cannot be dealt with well, in this paper we propose an objection generation model, which is composed of a LSTM decoder and SCDV of the user opinion, as shown in the lower part of Fig. 1. This part is described in Sect. 4.

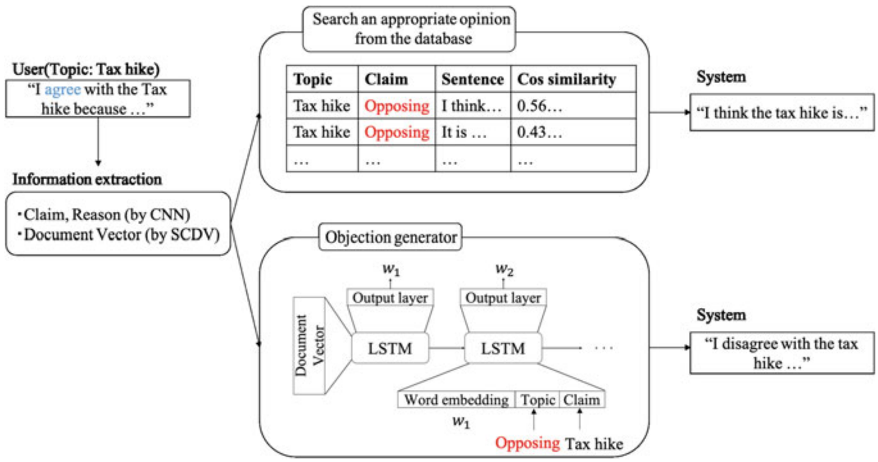


Fig. 1 Overview of the debate system

3 Debate Database and Motivation for Generating Objections

We employed Inoue's method [7] for collecting opinions from the Web, and created a database for debates. We collected 4 topics. The number of collected opinions is shown in Table 1.

Depending on the topic and claim, the number of opinions that can be collected is unbalanced in terms of supporting and opposing, as well as in terms of topics. Therefore, there is a possibility that the system cannot find the appropriate objection to the user's opinion because of the limited number of candidate opinions. In addition, even for a topic with a large number of collected opinions, the system may not be able to find the appropriate objection if there are no candidate objections that deal

Table 1 Debate Database Information

Topics	Claims	Number of opinions	Avg. length of words
Capital punishment	Supporting	228	60.84
	Opposing	283	64.19
Nuclear power plant	Supporting	124	64.06
	Opposing	320	80.68
Tax hike	Supporting	126	65.03
	Opposing	202	70.44
Casino bill	Supporting	52	57.40
	Opposing	141	87.20

with the point being made in the argument. Hence, in such a case, we propose a language model to generate more appropriate objections to the users' claim instead of selecting the candidate objections.

4 Model of Objection Generator

The structure of the language model for our objection generator is shown in Fig. 2, which corresponds to the lower part of Fig. 1. In recent years, models have been proposed that concatenate additional information to the input layer and generate a characteristic response sentence [8, 9]. We applied these models to our objection generator and employed the topic and claim as additional information. Here, the "Topic" is capital punishment, a casino bill or the like, and the "Claim" is either supporting or opposing.

In training the model, we encode a document (opinion) $D = \{w_0, w_1, ..., w_n\}$, into a vector representation DV created by SCDV (Due to space limitations, the SCDV explanation is omitted here. Please refer to [6] for an explanation.). Then, the original opinion sentence is reproduced by LSTM [3] from the DV as shown in Fig. 2. The values of hidden units in LSTM are obtained by combining the value of hidden units produced at the previous time step $t-1$, the word representations e_t at the current time step t, the topic vector t_i for topic $i\,(i \in [0, 1, 2, 3])$, and the claim vector c_j for claim $j\,(j \in [0, 1])$. An input gate, a memory gate, and an output gate, respectively denoted as i_t, f_t and o_t, are computed as follows:

$$\begin{bmatrix} i_0 \\ f_0 \\ o_0 \\ l_0 \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} W_{DV} \cdot SCDV(D) \tag{1}$$

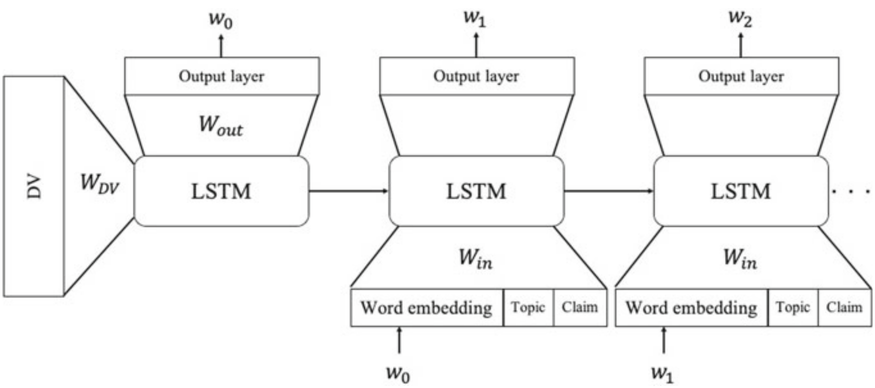


Fig. 2 Objection generator

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ l_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} W_{in} \cdot \begin{bmatrix} h_{t-1} \\ e_t \\ t_i \\ c_j \end{bmatrix} (t \geq 1) \tag{2}$$

Each value of the topic vector $t_i (i \in [0, 1, 2, 3])$ and claim vector $c_j (j \in [0, 1])$ is randomly initialized by the value drawn independently from Gaussian distribution $N(0, 1)$. After training and generating an objection, the claim vector is replaced with the desired one, and the model predicts w_t by computing the following equation iteratively: $w_t = \texttt{softmax}(W_{out}(h_t))$. See the related work [9] for the computation of h_t.

5 Experiments

The human evaluation and BLEU were conducted using 90% of the collected opinions (Table 1) as training data and the rest as test data. We used 1-layer LSTM models with 180 hidden cells. The word embedding size, topic vector size, claim vector size, and vocabulary size were 256, 32, 50, and 7,557, respectively. The optimization method was Adam [10] and the Document Vector DV size was 2,000.

5.1 BLEU

In this experiment, we clarify the sentence generation ability from document vectors and the effect of the topic vector and claim vector. The BLEU score [12] was calculated between the original sentence and the sentence generated from the document vector, which was converted from the original sentence. The results are shown in Table 2. BLEU-1, BLEU-2, BLEU-3, and BLEU-4 are 1-gram, 2-gram, 3-gram, and 4-gram precision, respectively. "Objection Generator" is a model that does not use the claim vector or topic vector. "Objection Generator-T" is a model that only uses the topic vector, and "Objection Generator-T&C" is a model that uses both. "LSTM Encoder-Decoder" [11] was implemented for comparison. "Objection Generator" shows better performance than LSTM Encoder-Decoder. The reason is that it is difficult to train the encoder because of limited data and the number of words per sentence is large. On the other hand, our model uses SCDV to reduce the number of parameters related to encoding. Moreover, the topic vector proved to be effective. Since the opinion vectors are well divided into each topic, it seems that SCDV helps our model to discriminatively learn the vocabulary used in each topic.

Table 2 BLEU score

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
LSTM Encoder-Decoder	3.71	1.07	0.31	0.00
Objection Generator	6.32	2.76	1.03	0.00
Objection Generator—T	9.52	3.70	1.36	0.46
Objection Generator—T&C	**10.01**	**3.94**	**1.51**	**0.63**

5.2 Human Evaluation

A 5-point Likert scale evaluation was conducted using the top 30 generated sentences with the highest cosine similarity for each input sentence. Comparison with human opinion sentences selected by the method described in Sect. 2 was conducted, and the result is shown in Fig. 3.

Ten participants evaluated generated or selected sentences on "Naturalness" (whether the wording is natural), "Clarity" (whether the claim [supporting or opposing] is clear), "Reason" (whether appropriate reason is included), "Viewpoint" (Whether the point of view is the same as the point of view of the user opinion sentence), and then the scores were averaged. Although our model is superior in

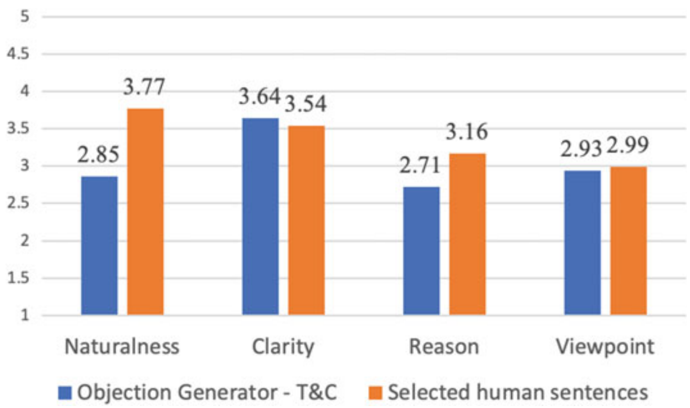


Fig. 3 Results on generated sentences having the highest cosine similarity

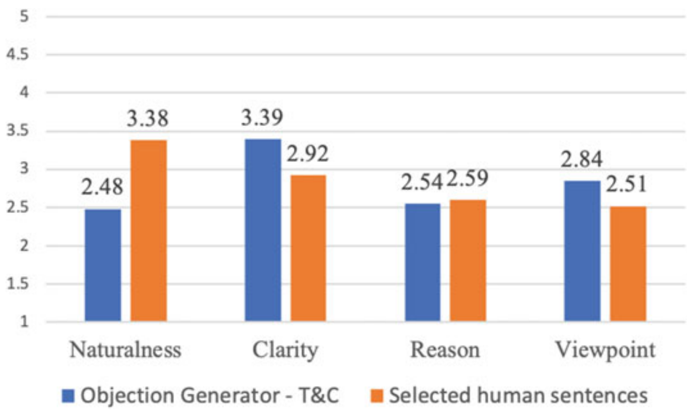


Fig. 4 Results on generated sentences having higher cosine similarity than selected human opinion

"clarity", it cannot be said that our performance is better than the selected human opinion sentences. Here, the averaged results for only 13 sentences generated with higher cosine similarity than the selected human opinion sentences are shown in Fig. 4. Since problems remain in "naturalness", our model is superior to selected human opinion sentences in "Clarity" and "Viewpoint".

6 Conclusions

We proposed a model that can generate objections based on the topic and claim vectors when there were no suitable candidate sentences. Experiments showed the effectiveness of this model. In particular, the model shows good performance in "clarity", which is considered to be owing to the effect of the Topic Vector. However, at this time, from the viewpoint of "reason", the sentences generated by our model are inferior to selected opinion sentences created by humans. We will endeavor or improve our model in this regard in future research. In addition, there is also a problem regarding "naturalness", so that it is necessary to consider techniques (such as pre-training) using data other than opinions collected. Finally, regarding the human evaluation, examples of generated sentences that obtain good evaluation are shown in Table 3.

Table 3 Examples of generated objections

Inputs	Generated Objections
「誰でも良かった」等と発言する犯罪者もいますが、それでも何らかの感情があるから殺しているんだと思います。更生する可能性があるのだとしたら、死刑制度には反対です。死刑にしなくても、刑務所の中で働いたりして反省するかもしれないので、殺人を犯した人は終身刑でもいいのではないかと思います。 (Some criminals say, “It doesn’t matter who it was,”, but I still think that they killed people because there was some motivation. If they could be rehabilitated, I would oppose the capital punishment. Since they may be rehabilitated by receiving punishment in the prison, I think that it is better for them to receive life imprisonment rather than the depth penalty.)	死刑に賛成です。親兄弟が皆殺しにされて、終身刑を最高刑にする事はおかしいと思います。殺された遺族の方は刑務所生活の生活費が税金とかなるの？ (I agree with capital punishment. I think that it is wrong that the punishment of the criminal who killed my family is, at most, only life imprisonment. How do the bereaved families feel about the fact that the cost of keeping the criminal in prison all that time is paid for by their taxes?)
私は死刑には大反対です。なぜなら、どんな理由があろうと、国家が殺人を犯すのはおかしいから。死刑という名で殺人を国家がしていいわけがない。 (I am very mush against capital punishment because, whatever the reason, it is wrong for the state to commit murder. Even the capital punishment system does not mean that the state can murder criminals.)	死刑に賛成です。死刑に値する罪を犯した人の世話を税金で養う事に疑問があるのです。犯人を奪ったんです。 (I agree with capital punishment. I question using tax money to take care of those who committed serious crimes. Capital punishment removes the criminal and the cost of imprisoning him/her.)
私は死刑制度に賛成します。人を何人殺しても警察に捕まり裁判にかけられても、命の保証はあります。また社会復帰の道もあります。 (I agree with the capital punishment system. If there is no capital punishment system, then even if a person kills many people, gets caught by the police and is tried and convicted, he will still live. He might even make it back into society one day.)	私は死刑制度に反対です。理由は、冤罪の可能性を減らすために。国家権力によって人を殺す権利があるだろうか。 (I disagree with capital punishment we need to reduce the possibility of false charges. The government has no right to kill people)
私は死刑制度には賛成です。罪に応じた刑罰が必要で、被害者の社会復帰無くして加害者の更生などは意味がないと考えているからです。... (長文のため省略) ... 人を殺めたらやはりそれは厳正な裁判の結果、死をもって償うべきであると思っています。 (I agree with the capital punishment system. This is because punishment is necessary for sin, and since the victim cannot return to society, it seems meaningless to rehabilitate the perpetrator. (An Omission) If you kill a person, I think you should be judged severely and pay for your crime with your death.)	私は死刑制度に反対です。理由は、冤罪の可能性を否定できるのですか。 (I disagree with capital punishment. The reason is, can the possibility of false charges be denied?)
実のところ、私もカジノには反対だ。カジノが日本人のためになるとは思えない。しかし、カジノは隔離された場所で行われるものである。 (Actually, I am against the casino as well. I do not think that casinos will benefit the Japanese. However, the casino is in an isolated place.)	実は、カジノ大賛成です。日本人の博打好きはたいへん多い。 (Actually, I am in favor of the casino. There are a lot of Japanese fans like gambling.)

References

1. Yoshino K, Kawahara T (2015) News navigation system based on proactive dialogue strategy. In: IWSDS
2. Zhao T, Lu A, Lee K, Eskenazi M (2017) Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. In: Proceedings of SIGDIAL, pp 27–36
3. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
4. Marumoto R, Tanaka K, Takiguchi T, Ariki Y (2018) Debate dialog for news question answering system 'Nettv ' -debate based on claim and reason estimation-. In: IWSDS
5. Shi H, Ushio T, Endo M, Yamagami K, Horiii N (2016) A multichannel convolutional neural network for cross-language dialog state tracking. In: IEEE workshop on spoken language technology
6. Mekala D, Gupta V, Paranjape B, Karnick H (2017) SCDV: sparse composite document vectors using soft clustering over distributional representations. In: Proceedings of EMNLP
7. Inoue Y, Fujii A (2009) A Web mining system for points at issues of current topics. DEIM Formu 2009:i1–25 (in Japanese)
8. Li J, Galley M, Brockett C, Spithourakis G, Gao J, Dolan B (2016) A persona-based neural conversation model. In: Proceedings of ACL
9. Luan Y, Brockett C, Dolan B, Gao J, Galley M (2017) Multi-task learning for speaker-role adaptation in neural conversation models. In: Proceedings of IJCNLP
10. Kingma D, Ba J (2015) Adam: a method for stochastic optimization. In: Proceedings ICLR
11. Vinyals O, Le Q (2015) A neural conversational model. In: Proceedings of ICML deep learning work-shop
12. Papineni K, Roukos S, Ward T, Zhu W (2002) BLEU: a method for automatic evaluation of machine translation. In: Proceedings of ACL
13. Maaten L, Hinton G (2002) Visualizing data using t-SNE. In: Proceedings JMLR, pp 2579–2605

A Differentiable Generative Adversarial Network for Open Domain Dialogue

Asier López Zorrilla, Mikel deVelasco Vázquez, and M. Inés Torres

Abstract This work presents a novel methodology to train open domain neural dialogue systems within the framework of Generative Adversarial Networks with gradient based optimization methods. We avoid the non-differentiability related to text-generating networks approximating the word vector corresponding to each generated token via a *top-k softmax*. We show that a weighted average of the word vectors of the most probable tokens computed from the probabilities resulting of the top-k softmax leads to a good approximation of the word vector of the generated token. Finally we demonstrate through a human evaluation process that training a neural dialogue system via adversarial learning with this method successfully discourages it from producing generic responses. Instead it tends to produce more informative and variate ones.

Keywords Dialogue systems · Generative adversarial networks · Open domain dialogue

1 Introduction

Open domain dialogue systems or chatbots are systems deployed to interact with humans offering coherent responses according to the dialogue history. Unlike task-oriented dialogue systems, there is no specific goal to be achieved during the inter-

A. López Zorrilla (✉) · M. deVelasco Vázquez · M. I. Torres
Faculty of Science and Technology, University of the Basque Country UPV/EHU, Sarriena s/n, 48940 Leioa, Spain
e-mail: asier.lopezz@ehu.eus

M. deVelasco Vázquez
e-mail: mikel.develasco@ehu.eus

M. I. Torres
e-mail: manes.torres@ehu.eus

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_24

action by the system. The only goal is to generate appropriate, relevant, meaningful and human-like utterances.

This area of research has gained an increasing amount of interest from the community since the advent of sequence-to-sequence neural network models [22]. These neural networks are capable of processing and generating sequences of data of arbitrary length, which makes them very suitable for this research [21, 24]. The task of open domain dialogue generation can easily be cast as a sequence transduction problem, where the input is the sequence of words corresponding to the last user's utterance, and the output are the words of the system's response. It is also possible to condition the output of the network to a broader dialogue context or other knowledge sources in order to increase the coherence of the responses [6, 19], but in this work we will not research in that direction.

These neural models are usually learnt from corpora composed of input utterance-response pairs, via supervised learning. Movies subtitles, Twitter or online forums can be used as the source of these data. In this framework, the neural network is trained to minimize a distance between the generated response and the desired one. Even though interesting performances can be obtained with this procedure, it frequently yields models that tend to generate dull and safe responses which appear frequently in the corpus, such as *I don't know* or *I'm sorry*.

We build upon Generative Adversarial Networks (GANs) [7] to overcome this problem and to increase the overall variety in the responses of the neural dialogue model, as these have shown promising results in many data generation tasks. While in supervised learning a unique desired output is assigned to each input in the corpus, GANs allow many correct outputs, which makes much more sense in dialogue, and models better the one-to-many property of input-output pairs [23]. The learning methodology for GANs involves training two neural networks, a generator and a discriminator, in an adversarial fashion. The generator tries to learn a data distribution while the discriminator learns whether a given sample corresponds to the training data or has been generated by the generator. In the context of dialogue systems, the generator would be the sequence-to-sequence model and the discriminator would act as a Turing Test.

GANs were first successful in image generation tasks. More recently text-related problems, such as machine translation [25], text generation [26, 27] or image captioning [20] have also been tackled within this framework. GANs have also been applied in the research of dialogue systems, yet only on a few occasions. References [5, 11] experiment with training discriminators that could measure the quality of the utterances generated by chatbots. On the other hand [9, 14] go a step further and train neural dialogue systems via adversarial learning, but with the drawback that they make use of reinforcement learning instead of gradient-based optimization methods. This is due to text being represented as a sequence of discrete tokens, which breaks the differentiability of the discriminator's output with respect to the generator's parameters, as explained in Sect. 3.

In this context, the contributions of our work are twofold. First, we present a novel methodology to avoid this non-differentiability: the *top-k softmax*. Since the top-k softmax allows to plug-in the output of the generator into the discriminator in a differentiable manner, our approach is simpler and easier to implement than other dialogue systems trained in the GAN framework. Second, we demonstrate that training a neural dialogue system via adversarial learning with this method successfully discourages it from producing generic responses, and that it often leads to more informative responses too.

The rest of the paper is organized as follows. In Sect. 2 we specify the chosen architecture for the sequence-to-sequence dialogue model and the baseline training procedure. In Sect. 3 we describe the proposed GAN for dialogue generation based on the top-k softmax and compare it to alternative approaches to deal with the differentiability problem. In Sect. 4 we give all the details about our experimental setup and hyper-parameter choice. Section 5 shows the results of two experiments to validate our proposal. We conclude with the final remarks in Sect. 6.

2 Sequence-to-Sequence Dialogue Model Architecture

The chosen architecture for the dialogue model is a standard sequence-to-sequence network with attention [1]. Given an input sequence of length T of discrete integer tokens $x = x_1, x_2, ..., x_T$, the corresponding sequence of vectorial word representations $\mathbf{v} = \mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_T$ can be obtained via the word vector matrix $\mathbf{W}$, just by taking the corresponding row $\mathbf{v}_i = \mathbf{W}[x_i]$ per each token x_i. The size of $\mathbf{W}$ is $V \times D$, where V is the vocabulary size and D the dimension of each word vector. The encoder takes this sequence of vectors and produces another sequence of vectors of the same length $\mathbf{h} = \mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T = \text{encoder}(\mathbf{v})$. In our work the encoder is a deep bidirectional Long Short Term Memory (LSTM) Recurrent Neural Network (RNN).

To proceed with the generation of the output sequence $y = y_1, y_2, ..., y_\tau$, a global attention mechanism is applied as in [17]. At the time step t of the generation, the decoder is fed with the discrete integer token generated at previous time step, y_{t-1}. Then the corresponding word vector $\mathbf{W}[y_{t-1}]$ is input to the decoder's RNN and this outputs $\mathbf{o}_t$. Of course, due to the architecture of RNNs, $\mathbf{o}_t$ is conditioned, though implicitly, not only to y_{t-1} but also to all the previously generated tokens. In our experiments this neural network is also a deep LSTM. $\mathbf{o}_t$ is then transformed to $\tilde{\mathbf{o}}_t$ via a multilayer perceptron (MLP) that takes as input o_t and also $\mathbf{c}_t$, the context-vector produced by the attention mechanism at time step t. $\mathbf{c}_t$ is a weighted average of the encoder's output vectors:

$$\mathbf{c}_t = \sum_{j=1}^{T} a_{jt} \mathbf{h}_j \ , \tag{1}$$

where a_{jt} is the score between $\mathbf{h}_j$ and $\mathbf{o}_t$, i.e., how much attention should be put on the output of the encoder at the encoding time step j on the time step t of the decoding phase. a_{jt} is a softmax-normalized scalar output of another MLP, that takes

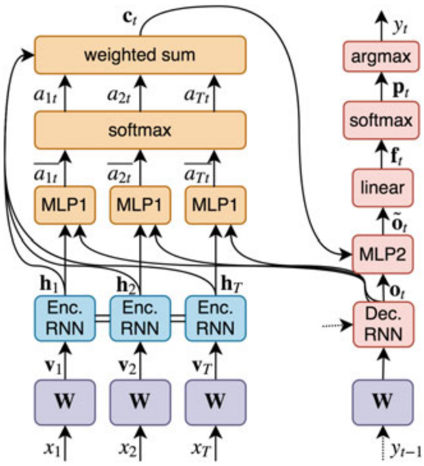


Fig. 1 A diagram of the chosen sequence-to-sequence network: blue transformations refer to the encoder, orange to the attention mechanism, purple to the word matrix (shared between the encoder and decoder), and red to the decoder. For simplicity, only the time step t of the decoding is shown

as input $\mathbf{h}_j$ and $\mathbf{o}_t$, and outputs $\overline{a_{jt}}$. With the softmax normalization we ensure that all the scores at time step t are positive and sum one:

$$a_{jt} = \frac{\exp\left(\overline{a_{jt}}\right)}{\sum_{j'=1}^{T} \exp(\overline{a_{j't}})} \tag{2}$$

Finally, $\tilde{\mathbf{o}}_t$ is linearly projected to a vector of dimension V: $\mathbf{f}_t = \text{linear}(\tilde{\mathbf{o}}_t)$. This vector represents an unnormalized probability distribution over all the possible words in the vocabulary. A softmax normalization is then applied to $\mathbf{f}_t$ to get $\mathbf{p}_t = \text{softmax}(\mathbf{f}_t)$, the normalized version of $\mathbf{f}_t$. The output token at time step t, y_t, can be sampled from $\mathbf{p}_t$ taking the argument of the maxima:

$$y_t = \arg\max_i \left(\mathbf{p}_t[i]\right) \tag{3}$$

Generation stops at time τ, when y_τ corresponds to the end-of-sequence token. The architecture of the network is summarized in Fig. 1.

Maximum Likelihood Estimation via Supervised Learning As aforementioned, this neural network can be trained from a corpus composed of input-output sequence pairs via supervised learning. A maximum likelihood estimation (MLE) of the parameters of the network can be carried out by minimizing the word level cross entropy loss L_{MLE}:

$$L_{MLE} = \frac{1}{|\mathcal{C}|} \sum_{x,s \in \mathcal{C}} \frac{1}{|s|} \sum_{t=1}^{|s|} -\log \mathbf{p}_t[s_t] \ , \tag{4}$$

where $\mathcal{C}$ is a corpus composed of pairs of inputs x and desired outputs s, s_t each of the words in s, and $\mathbf{p}_t[s_t]$ the output of the network in the t-th time step corresponding to the token s_t. We omit the output's dependence on x to keep the notation simple.

During training we employ the teacher forcing strategy, i.e., in the t-th step of the decoding we feed the ground true token s_{t-1} to the decoder's RNN instead of the prediction y_{t-1}. We experimented with other sampling techniques such as scheduled sampling [2], but we found no improvement.

3 Sequence Generative Adversarial Network Training

In the context of dialogue systems, the generator network in the GAN is the sequence-to-sequence dialogue model, which produces a response y to the input utterance x. The discriminator is another network that acts like a Turing Test: it takes an input utterance x and a response r as inputs, and outputs a scalar between 0 and 1 representing the network's confidence level on r being produced by a chatbot. Namely, the lower the output of the discriminator is, the more human-like r is according to the discriminator's criteria.

The procedure to train the dialogue system in this framework involves iteratively updating the generator and the discriminator. The generator is trained to fool the discriminator and make it think that its responses are human-like, and in contrast the discriminator is trained to distinguish between human and bot responses.

Let us now define the losses to be minimized in this two optimization procedures. Given a batch of input utterances, responses and labels indicating whether each response has been generated by a bot or a human, the discriminator's parameters will be updated to minimize the next cross-entropy loss:

$$L_D = \frac{1}{|\mathcal{B}_D|} \sum_{x,r,l \in \mathcal{B}_D} -\left[l \cdot \log a + (1-l) \cdot \log(1-a)\right] , \tag{5}$$

where $\mathcal{B}_D$ is a batch composed of tuples of input utterances x, responses r and boolean labels l, and a the output of the network given x and r.

The objective for the generator is just to minimize the output of the discriminator when the latter is fed with a batch of input utterances and the responses of the generator to those same input utterances:

$$L_G = \frac{1}{|\mathcal{B}_G|} \sum_{x \in \mathcal{B}_G} a , \tag{6}$$

where $\mathcal{B}_G$ is a batch composed of input utterances x. a is the output of the discriminator given x and y, where y is the output of the generator given x.

The differentiability problem We have already described the architecture of the generator in Sect. 2. On the other hand, the discriminator is a composed of two deep bidirectional LSTM-RNNs, for x and r respectively, followed by some fully-connected layers. Before being processed by the RNNs, both x and r integer sequences are converted to word vector sequences via the same word vector matrix $\mathbf{W}$, as explained in Sect. 2.

Being these the network architectures, it is not possible to differentiate L_G (Eq. 6) with respect to the parameters of the generator. The problem arises with the argmax operation in the sequence of transformations that converts $\mathbf{f}_t$ into $\mathbf{u}_t$:

$$\mathbf{f}_t \underset{\leftarrow}{\xrightarrow{\text{softmax}}} \mathbf{p}_t \underset{\not\leftarrow}{\xrightarrow{\text{argmax}}} y_t \underset{\leftarrow}{\xrightarrow{\mathbf{W}[y_t]}} \mathbf{u}_t \ , \tag{7}$$

where $\mathbf{f}_t$ is the unnormalized probability distribution over all the possible words in the vocabulary in the step t of the generation, $\mathbf{p}_t$ the softmax-normalized version of $\mathbf{f}_t$, y_t the argument of the maxima of $\mathbf{p}_t$, and $\mathbf{u}_t$ is the word vector corresponding to the token y_t. Green arrows indicate that the operation is differentiable, whereas red arrows that it is not.

The top-k softmax We propose a novel alternative computation path that approximates $\mathbf{u}_t$ in a fully differentiable manner, allowing the generator to be trained with very convenient gradient-based methods. The idea behind this path is to generate a word vector $\tilde{\mathbf{u}}_t$, hopefully similar to $\mathbf{u}_t$, as a weighted average over the word vectors corresponding to the k most probable words according to $\mathbf{f}_t$. $k \geq 2$ is an integer parameter of the transformation. In short, the differentiable computation path is as follows:

$$\mathbf{f}_t \underset{\leftarrow}{\xrightarrow{\text{top-k}}} \mathbf{k}_t, \tilde{\mathbf{f}}_t \underset{\leftarrow}{\xrightarrow{\text{softmax}}} \mathbf{k}_t, \tilde{\mathbf{p}}_t \underset{\leftarrow}{\xrightarrow{\sum_i \tilde{\mathbf{p}}_t[i]\cdot\mathbf{W}[\mathbf{k}_t[i]]}} \tilde{\mathbf{u}}_t \tag{8}$$

The first operation in Eq. 8 performs a selection of the *top-k* elements in $\mathbf{f}_t$. It outputs $\mathbf{k}_t$ and $\tilde{\mathbf{f}}_t$. $\mathbf{k}_t$ are the indices corresponding to the k elements in $\mathbf{f}_t$ with the highest values, and $\tilde{\mathbf{f}}_t$ are those values. In other words, $\mathbf{k}_t$ represents the most probable words, and $\tilde{\mathbf{f}}_t$ their unnormalized probabilities. The second operation is just a softmax normalization of these k probabilities. It converts $\tilde{\mathbf{f}}_t$ into $\tilde{\mathbf{p}}_t$. See Fig. 2 for a graphical example. Finally, the approximated word vector that will be fed to the discriminator's RNN is computed as the weighted average of the word vectors corresponding to tokens $\mathbf{k}_t$, where the weights are the probabilities $\tilde{\mathbf{p}}_t$:

$$\tilde{\mathbf{u}}_t = \sum_{i=1}^{k} \tilde{\mathbf{p}}_t[i] \cdot \mathbf{W}[\mathbf{k}_t[i]] \tag{9}$$

Note that in the whole process the differentiability has not been broken. Therefore, and in contrary to the previous computation path (Eq. 7), the partial derivatives of $\tilde{\mathbf{u}}_t$ with respect to $\tilde{\mathbf{f}}_t$ exist and are non zero. In Sect. 5 we show that $\tilde{\mathbf{u}}_t$ is a good approximation of $\mathbf{u}_t$ when k is small. In fact, $\mathbf{u}_t$ is the nearest neighbor of $\tilde{\mathbf{u}}_t$ the 98% of the times with $k = 2$.

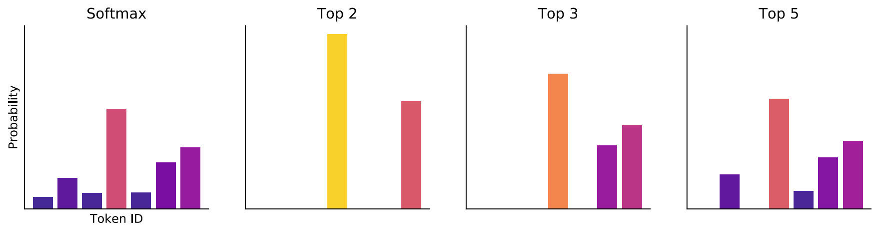


Fig. 2 On the left, a graphical example of a softmax normalization of a $\mathbf{f}_t$ distribution. The rest of the plots show the top-k softmax normalizations of $\mathbf{f}_t$ for different values of k

Related approaches Before continuing with our proposal for training the GAN, let us briefly compare the top-k softmax with alternative approaches to deal with the non-differentiable argmax operation. Apart from the aforementioned reinforcement learning-related methodologies based on [27], we are only aware of works [13] that in one way or another tackle this problem with the concrete or Gumbel-softmax distribution [10, 18]. This is a continuous relaxation of discrete random variables. In short, it transforms a probability distribution into a relaxed one-hot vector corresponding to a randomly taken sample from that distribution. That relaxed vector is different from the result of the top-k softmax in two important aspects. First, it is non-deterministic, which could be interesting but also unnecessary for our application. Second, all its elements are non-zero, which means that approximating a word vector as a weighted average according to those probabilities would imply mixing all the word vectors in the vocabulary, which seems again inadequate for our application.

A discrete version of this transformation is the Straight-Through Gumbel-softmax estimator [3, 10], which was used by [16, 20]. It serves to approximate the gradients of a one hot vector sampled according to a probability distribution. Thus it avoids the problem of averaging over all the word vectors, but it is still non-deterministic. Moreover, the operation is still non-differentiable. Even though this method provides an estimation of the gradients in this scenario, but using it could be risky because it might cause discrepancies between the forward and backward passes, as stated in the original work [10].

Training procedure The top-k softmax allows L_D to be differentiable with respect to the parameters of the generator. Thus gradient-based optimization methods can be applied to train both the generator and the discriminator. Let us now specify the general training loop and the pretraining strategies applied in this work.

Prior to the training of the dialogue system, we pretrain the word vector matrix in the same corpus that will be used later. Following the work of [9, 14], we also pretrain the generator using the MLE criteria, and the discriminator with the responses generated by the pretrained generator and with responses from the corpus. In order to stabilize the rest of the training process and to avoid the catastrophic forgetting phenomenon of the discriminator, each time we sample a response of the generator to a given input, we add it to a corpus of generator's turns $\mathcal{C}_D$.

Now we enter the main training loop, where the generator and the discriminator will be trained adversarially. This loop will be run for many iterations. We start it training the generator to minimize the output of the discriminator according to Eq. 6 during a number of iterations. Then we increase the corpus $\mathcal{C}_D$ with the current state of the generator, and train the discriminator during another number of iterations. More recent input-response pairs are taken with a higher probability than the older ones from $\mathcal{C}_D$ when training the discriminator.

We finally repeat this process of training the generator, adding samples to $\mathcal{C}_D$ and training the discriminator, but this time training the corpus with the MLE criteria. This approach is also taken in [9, 14], and it aims at stabilizing the training process. In order to further stabilize it, we reduce the learning rate of the training optimizer throughout the global iterations.

This whole procedure is summarized in the Algorithm 1.

Algorithm 1 An adversarial training strategy for neural dialogue models

Require: Generator G, Discriminator D, Corpus $\mathcal{C}$, training hyper-parameters.
Pretrain word vector matrix **W** on $\mathcal{C}$.
Pretrain G minimizing L_{MLE} (Eq. 4).
Initialize $\mathcal{C}_D$ with G's responses y to some inputs x.
Pretrain D minimizing L_D (Eq. 5).

for the number of total iterations, and with a decaying learning rate **do**
    Update G minimizing L_G on inputs x in $\mathcal{C}$ (Eq. 6).
    Add (x, y) pairs to $\mathcal{C}_D$ using G.
    Update D minimizing L_D.
    Update G minimizing L_{MLE} on $\mathcal{C}$.
    Add (x, y) pairs to $\mathcal{C}_D$ using G.
    Update D minimizing L_D.

4 Experimental Setup

All the experiments in this work were carried out with the OpenSubtitles2018 corpus [15], which is composed of around 400M utterances from movie subtitles. As proposed in [24], since the turns are not clearly indicated, we treat each utterance as the desired output for the previous one.

As for the text preprocessing, we removed some symbols and converted all the names, numbers and places to tags *<person>*, *<number>* and *<place>*, respectively. This was done with the Spacy entity recognizer [8]. Finally we defined the vocabulary with most 30000 frequent words, and deleted every other token from the corpus. We pretrained 300 dimensional word vectors of those tokens on the corpus, with FastText [4]. These are then optimized again throughout the training process.

Let us now give details about the architecture of the sequence-to-sequence generator. The deep bidirectional RNN encoder is made of two LSTM networks (one per direction) of 4 layers, 512 cells each. On the other hand, the decoder's LSTM has 4 layers of 1028 cells. The MLP that converts $\mathbf{o}_t$ and $\mathbf{c}_t$ into $\tilde{\mathbf{o}}_t$ (see Sect. 2 for more details) has one *leaky*-ReLU layer. The size of $\tilde{\mathbf{o}}_t$ is 500. The MLP that computes the attention score has two layers. The first one is a 250-sized hyperbolic tangent layer, and the second is a linear output layer that computes the scalar score.

Regarding the discriminator, its two deep bidirectional encoders share the same architecture: two LSTM networks of two layers, 128 cells each. This vector is then fed to a MLP of two layers: a *leaky*-ReLU layer of size 100 followed by a single sigmoidal unit. The chosen value for the k parameter of the top-k softmax was 2.

The most promising hyper-parameters we have found for the training procedure are summarized next. First of all, we used the Adam optimizer [12] with batch size of 256 throughout all the optimization processes. We pretrained the generator during 50000 training iterations with a fixed learning rate of 0.001. We sampled 125000 responses from that generator and then pretrained the discriminator during 1000 iterations, with the same learning rate. All the batches fed to the discriminator were balanced: there was a human example per each generator's example. Human and generator's example were uncorrelated; they did not share the input.

The main iteration loop was run 200 times. The initial learning rate was 0.001 with a decaying factor of 0.995 when training the discriminator and the generator with the MLE criteria. It was ten times smaller when training the generator to minimize the output of the discriminator. Every MLE step was run during 50 iterations, and every step of minimizing the discriminator's output was run during 35 iterations. After each of these steps, 5000 input-response pairs were sampled from the generator, and the discriminator was trained during 40 iterations.

It is worth mentioning that we did not vary the architectural hyper-parameters much during our experiments. They are similar to many other sequence-to-sequence networks in the literature. On the other hand, selecting good and stable training hyper-parameters is challenging. This requires a deeper and more specific research that we leave for future work.

5 Experiments

We now present an experimental validation of the proposed differentiable sequence generative adversarial network for dialogue generation in two series of experiments. First we validate our differentiable GAN architecture measuring the quality of the word vectors obtained with after the top-k softmax computation path presented in Eq. 8. Additionally, we compare a neural dialogue trained with this computation path and with the adversarial learning procedure summarized in the Algorithm 1 with a standard MLE model.

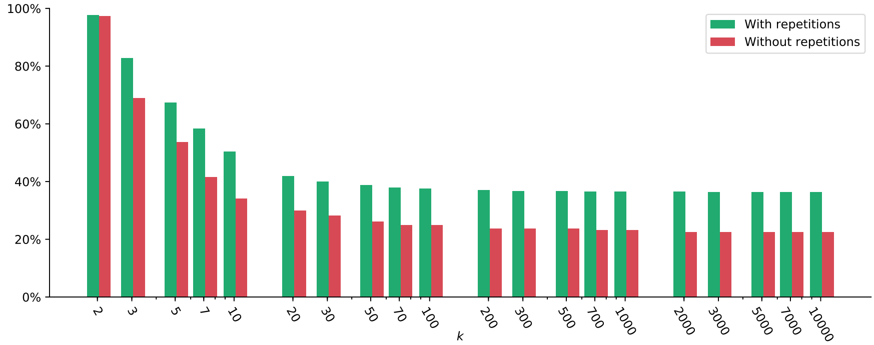


Fig. 3 Number of times that the actual word vector is the nearest neighbor of an approximated word vector produced by the top-k softmax, for different values of k

Table 1 Some responses to the same inputs in four different stages of the training procedure. *A* is the response of the pretrained MLE model. *B* is a response of an intermediate model just after a MLE step. *C* is the response of the same model than *B*, but just after a step of minimizing the output of the discriminator. *D* is the output of the final model

Input utterance	Response
Hi!	**A**: I'm <person> <person>. **B**: hi. **C**: hi, <person>. **D**: hello, <person>.
I love you.	**A**: I love you, too. **B**: I love you **C**: I love you too, <person> **D**: you know, I think maybe we could talk...
Who is she?	**A**: I don' know. **B**: she's my friend **C**: you'll never regret again. **D**: she's my girlfriend, <person>, and...
What's the color of the sky?	**A**: it's a blue light. **B**: the blue blue. **C**: it's the red of the moon. **D**: it's a beautiful blue sky
See you	**A**: <person> <person> **B**: I'm sorry, I'm sorry **C**: I'm sorry, I didn't mean to interrupt you **D**: see you later, <person>

Approximated word vectors We fed 1000 random inputs from the corpus to the dialogue system, and computed which was the closest word vector to each approximated one according to the euclidean distance, for different values of k. With $k = 2$, the closest word vector was the correct one the 98% of the times if we consider all the produced tokens, and the 97% if we do not consider repetitions. This two percentages decrease to 83%/69% respectively with $k = 3$, and to 74%/60% with $k = 4$. Figure 3 shows this statistic for more values of k. We therefore conclude that the proposed method to make the output of the discriminator differentiable with respect to the generator's parameters is appropriate, at least with $k = 2$.

Comparison between the MLE baseline and the GAN Let us show a preliminary comparison between the pretrained MLE dialogue model with the final system after the adversarial learning. We asked 10 human evaluators to interact freely with the two systems during some few minutes, which resulted in dialogues of 25 turns on average. Then they were asked to decide which of them was better in terms of (1) the variety of the responses, (2) coherence and (3) informativeness. 7 out of the 10 evaluators opined that the final system was more variate and informative, and there was a draw in terms of coherence.

This can also be seen in Table 1. It shows responses to the same inputs in different stages of the training procedure. Not only are the baseline and final models compared in the table, but it also lets us gain an insight into the short-term effect of each of generator's minimizing the output of the discriminator. It tends to complex and enrich the model's responses, sometimes at the cost of losing some coherence.

6 Conclusion

We have presented a novel methodology to allow text generating models be trained in the GAN framework with gradient based optimization methods, the top-k softmax, and we have validated it in the open domain dialogue generation task. We have shown that good approximations of the word vector corresponding to each of the tokens generated by the dialogue system can be obtained with the top-k softmax. Moreover, we have demonstrated through a human evaluation process that a dialogue model trained in these conditions produces more variate and informative responses than the baseline MLE model, while being as coherent as it. Ultimately, the intersection between dialogue systems and GANs is a very promising area of research. We expect many more ideas from the two fields will be combined, and that many more applications of the GANs in the dialogue research will arise.

Acknowledgements This work has been partially funded by the Basque Government under grant PRE_2017_1_0357, by the University of the Basque Country UPV/EHU under grant PIF17/310, and by the H2020 RIA EMPATHIC (Grant N: 769872).

References

1. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:14090473
2. Bengio S, Vinyals O, Jaitly N, Shazeer N (2015) Scheduled sampling for sequence prediction with recurrent neural networks. In: Advances in neural information processing systems, pp 1171–1179
3. Bengio Y, Léonard N, Courville A (2013) Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:13083432
4. Bojanowski P, Grave E, Joulin A, Mikolov T (2016) Enriching word vectors with subword information. arXiv preprint arXiv:160704606
5. Bowman SR, Vilnis L, Vinyals O, Dai AM, Jozefowicz R, Bengio S (2015) Generating sentences from a continuous space. arXiv preprint arXiv:151106349
6. Ghazvininejad M, Brockett C, Chang MW, Dolan B, Gao J, Yih W, Galley M (2017) A knowledge-grounded neural conversation model. arXiv preprint arXiv:170201932
7. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
8. Honnibal M, Montani I (2017) Spacy 2: natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. To appear
9. Hori T, Wang W, Koji Y, Hori C, Harsham B, Hershey JR (2019) Adversarial training and decoding strategies for end-to-end neural conversation models. Comput Speech Lang 54:122–139
10. Jang E, Gu S, Poole B (2016) Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:161101144
11. Kannan A, Vinyals O (2017) Adversarial evaluation of dialogue models. arXiv preprint arXiv:170108198
12. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:14126980
13. Kusner MJ, Hernández-Lobato JM (2016) Gans for sequences of discrete elements with the gumbel-softmax distribution. arXiv preprint arXiv:161104051
14. Li J, Monroe W, Shi T, Jean S, Ritter A, Jurafsky D (2017) Adversarial learning for neural dialogue generation. arXiv preprint arXiv:170106547
15. Lison P, Tiedemann J (2016) Opensubtitles2016: extracting large parallel corpora from movie and tv subtitles. European language resources association
16. Lu J, Kannan A, Yang J, Parikh D, Batra D (2017) Best of both worlds: transferring knowledge from discriminative learning to a generative visual dialog model. In: Advances in neural information processing systems, pp 314–324
17. Luong MT, Pham H, Manning CD (2015) Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:150804025
18. Maddison CJ, Mnih A, Teh YW (2016) The concrete distribution: a continuous relaxation of discrete random variables. arXiv preprint arXiv:161100712
19. Serban IV, Sordoni A, Bengio Y, Courville AC, Pineau J (2016) Building end-to-end dialogue systems using generative hierarchical neural network models. AAAI 16:3776–3784
20. Shetty R, Rohrbach M, Hendricks LA, Fritz M, Schiele B (2017) Speaking the same language: matching machine to human captions by adversarial training. In: Proceedings of the IEEE international conference on computer vision (ICCV)
21. Sordoni A, Galley M, Auli M, Brockett C, Ji Y, Mitchell M, Nie JY, Gao J, Dolan B (2015) A neural network approach to context-sensitive generation of conversation responses. arXiv preprint arXiv:150606714
22. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in neural information processing systems, pp 3104–3112
23. Tuan YL, Lee HY (2019) Improving conditional sequence generative adversarial networks by stepwise evaluation. IEEE/ACM Trans Audio, Speech, Lang Process

24. Vinyals O, Le Q (2015) A neural conversational model. arXiv preprint arXiv:150605869
25. Wu L, Xia Y, Zhao L, Tian F, Qin T, Lai J, Liu TY (2017) Adversarial neural machine translation. arXiv preprint arXiv:170406933
26. Xu J, Ren X, Lin J, Sun X (2018) Diversity-promoting gan: A cross-entropy based generative adversarial network for diversified text generation. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 3940–3949
27. Yu L, Zhang W, Wang J, Yu Y (2017) Seqgan: sequence generative adversarial nets with policy gradient. In: AAAI, pp 2852–2858

A Job Interview Dialogue System with Autonomous Android ERICA

Koji Inoue, Kohei Hara, Divesh Lala, Shizuka Nakamura, Katsuya Takanashi, and Tatsuya Kawahara

Abstract We demonstrate a job interview dialogue with the autonomous android ERICA which plays the role of an interviewer. Conventional job interview dialogue systems ask only pre-defined questions. The job interview system of ERICA generates follow-up questions based on the interviewee's response on the fly. The follow-up questions consist of two kinds of approaches: selection-based and keyword-based. The first type question is based on selection from a pre-defined question set, which can be used in many cases. The second type of question is based on a keyword extracted from the interviewee's response, which digs into the interviewee's response dynamically. These follow-up questions contribute to realizing natural and trained dialogue.

1 Introduction

Spoken dialogue systems have been developed for various scenarios so far such as smartphone apps and smart speakers, and future systems are expected to handle more social interaction like human-human dialogues in our daily lives. For example, in

K. Inoue (✉) · K. Hara · D. Lala · S. Nakamura · K. Takanashi · T. Kawahara
Graduate School of Informatics, Kyoto University, Kyoto, Japan
e-mail: inoue@sap.ist.i.kyoto-u.ac.jp

K. Hara
e-mail: hara@sap.ist.i.kyoto-u.ac.jp

D. Lala
e-mail: lala@sap.ist.i.kyoto-u.ac.jp

S. Nakamura
e-mail: shizuka@sap.ist.i.kyoto-u.ac.jp

K. Takanashi
e-mail: takanashi@sap.ist.i.kyoto-u.ac.jp

T. Kawahara
e-mail: kawahara@sap.ist.i.kyoto-u.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_25

a real-life job interview, interviewers make various kinds of questions directly to interviewees in order to elicit information necessary for determining acceptance. A spoken dialogue system is expected to play the role of an interviewer so that interviewees are able to practice answering against expected questions. To assist with interview practice, the system has to realize a job interview dialogue similar to real human-human dialogue.

We demonstrate a job interview dialogue with autonomous android ERICA [3, 4] in the role of an interviewer. ERICA looks like a human being and is able to generate various behaviors including non-verbal ones such as eye gaze and head nodding. Current spoken dialogue systems for job interview dialogue ask only pre-defined questions [1, 2, 6, 7]. Although it is important for interviewers to dynamically ask follow-up questions to know more about the current topic, only a small number of studies have been conducted on automatic generation of follow-up questions [8]. In this demonstration, ERICA generates follow-up questions based on how well the interviewee's response fulfills the previous question and also keywords extracted from that response. It is expected that these follow-up questions make the job interview dialogue more natural and stimulate some thought from the interviewee, which is required for a practical system for job interview training.

2 System Configuration

While the basic dialogue flow is controlled with finite state transition, questions are generated based on interviewees' responses as explained below.

2.1 Scenario

In this demonstration, ERICA plays the role of the interviewer in a job interview. The dialogue content is independent of any business category or company, so questions from ERICA focus on the motivation and experience of interviewees. Therefore, the proposed system can be applied to interviewees with various background without modifying the list of questions.

2.2 Dialogue Flow

The dialogue flow is controlled with finite state transitions as illustrated in Fig. 1. The current job interview consists of several topics. Example topics are *reasons for application* and *accomplishments in college*. Each topic starts with a base question which is followed by follow-up questions. The base question is an open question on the topic. For example, when the topic is *reasons for application*, a base ques-

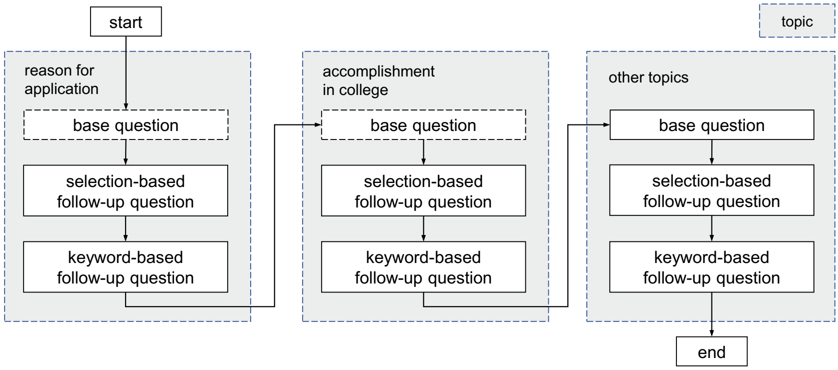


Fig. 1 Dialogue flow of job interview

tion is "What is the reason you applied for this job?" Topics and corresponding base questions are pre-defined. Next, based on an interviewee' response to the base question, the system generates a selection-based follow-up question. Finally, based on the response to the follow-up question, a keyword-based follow-up question is generated. Then, the current topic ends and proceeds to the next topic. The following section describes how to generate the above two kinds of follow-up questions in detail.

The other settings of ERICA's behaviors are as follows. ERICA generates head nodding when an interviewee is answering, based on a model for backchannel generation using prosodic features [5]. For the turn-taking behavior, ERICA waits for four seconds silence to take the floor. ERICA should wait for longer than in other dialogue scenarios because the system has to make sure that an interviewee has definitely finished answering. Note that ERICA generates head nodding while ERICA waits for the user's turn to end to avoid an unnatural silence. If an interviewee's response contains only a few nouns, ERICA will reply with a phrase such as "Sorry, could you say it again?" to prompt the interviewee to continue the answering.

2.3 *Generation of Follow-Up Questions*

We address how to generate the two kinds of follow-up questions below. At first, the system generates a selection-based follow-up question against a response to a base question. Next, a keyword-based follow-up question is generated against a response to the selection-based follow-up question.

2.3.1 Selection-Based Follow-Up Questions

We prepare several follow-up questions related to a base question, and the system selects an appropriate one based on the interviewee's response. This approach is robust against to any interviewee's responses because sentences of the questions are handcrafted. The prepared follow-up questions are categorized into three categories taking into account how well the response has fulfilled the base question.

- **category 1** (high priority) follow-up questions on which an interviewee must answer
- **category 2** (middle priority) follow-up questions on which an interviewee should answer
- **category 3** (low priority) other follow-up questions

Category 3 is a backup for categories 1 and 2. For example, for the topic *reason for apply*, follow-up questions can be as follows.

- **category 1** What is the specific reason why you want to work in our company?
- **category 2** What do you want to achieve after you enter our company?
- **category 3** Have you applied to any other companies?

The question of category 1 should be used when an interviewee's response did not satisfy the base question. The question of category 2 should be used when the interviewee accounted for the base question but did not mention in detail related to the follow-up question. The question of category 3 should be used when the interviewee has answered the base question well. Category 1 has the highest priority to be asked, followed by categories 2 and 3.

The system selects an appropriate follow-up question from the above-mentioned prepared set based on the degree of fulfillness of an interviewee's response. At first, if the number of nouns of an interviewee's response is larger than a threshold, follow-up questions of category 1 are excluded from candidates, since these were likely already mentioned in the interviewee's response. Next, for each follow-up question, the system checks if it has been addressed in the response. For this, in advance, we define a set of words to represent each follow-up question. The set contains words that appear in the sentence of the follow-up question itself. The set also include words related to the follow-up question. For example, related words for a follow-up question "*Where do you see yourself 10 years in the future?*" are *future*, *years*, and so on. Then, if one of the words in an interviewee's response to a base question is semantically close to one of the words in the above-mentioned list, the corresponding follow-up question is excluded from the candidates. In the above example, if an interviewee's response includes *future*, the follow-up question will not be selected. To measure the semantic distance between words, we use word embedding (word2vec) and its cosine distance. Finally, among follow-up questions left in the set of candidates, the system selects one of them from the category with the highest priority. If there are several possible candidates in the same category, the system randomly selects one question.

2.3.2 Keyword-Based Follow-Up Questions

The system also generates another type of follow-up question based on a keyword contained in the interviewee's response. Although the selection-based follow-up question can always be generated, their sentences are pre-defined. Therefore, it might be difficult for interviewees to feel that their responses are comprehended by the system. To deal with this issue, the system dynamically generates a follow-up question reflecting the response of an interviewee.

The system extracts a keyword from an interviewee's response and fills the keyword in a template. First, the system calculates the inverse document frequency (IDF) value that was trained with the text8 corpus[1] and the gensim software.[2] We hypothesize that an appropriate keyword is a rarity so that it has a higher IDF value. A keyword is determined as having the highest score of the IDF values. If several nouns are continuously observed, they are regarded as a compound noun, and then it is processed as one noun. To calculate an IDF value for a compound noun, the maximum value is used from several IDF values of consisting nouns.

After extracting a keyword, the system fills the keyword in a pre-defined template to generate a sentence of a follow-up question. For example, when an extracted keyword is *spoken dialogue systems*, the system generates a follow-up question as "*You mentioned spoken dialogue systems, so could you explain them in more detail?*".

3 Dialogue Example

A dialogue example is introduced as below.[3] The topic is *reasons for application*. Note that **U** and **S** represent utterances of a user and ERICA, respectively. Figure 2 shows a snapshot of a job interview with ERICA. We use a 16-channel microphone array to enhance the interviewee's speech for automatic speech recognition so that the interviewee can talk with ERICA without holding a microphone (hands-free).

S1 What is the reason why you applied for this job? (**base question**)
U1 I have learned artificial intelligence in my university. In this company, I can apply my skills of AI technologies to the job and can obtain new skills. This is the reason why I applied for this job.
S2 I see. What kind of effort or preparation do you make to achieve your goal? (**selection-based follow-up question**)
U2 I learn the latest technologies because this industry drastically changes. Recently, I am learning machine learning methods.
S3 I see. Could you explain about machine learning in detail? (**keyword-based follow-up question**)

[1] http://mattmahoney.net/dc/textdata.html.

[2] https://radimrehurek.com/gensim/models/tfidfmodel.html.

[3] Demo video is available at https://youtu.be/JpTlSVp2zx8.

Fig. 2 Snapshot of job interview dialogue with ERICA

U3 I am especially interested in deep learning. It is reported that deep learning allows us to train a complicated model that can be applied to various classification tasks and achieve higher accuracy than conventional approaches.
S4 I see. Thank you.

4 Conclusion

We have described a demonstration of a job interview dialogue with the autonomous android ERICA. The proposed system generates follow-up questions based on an interviewee's response. The follow-up questions consist of two kinds of approaches: selection-based and keyword-based. While the former is based on a pre-defined set of questions, the latter dynamically reflects an interviewee's response by extracting a keyword which was used. These follow-up questions contribute to realizing a natural job interview dialogue with ERICA.

Acknowledgements This work was supported by JST ERATO Ishiguro Symbiotic Human-Robot Interaction program (Grant Number JPMJER1401), Japan.

References

1. Ali MR, Crasta D, Jin L, Baretto A, Pachter J, Rogge RD, Hoque ME (2015) LISSA-Live interactive social skill assistance. In: ACII, pp 173–179
2. Hoque ME, Courgeon M, Martin JC, Mutlu B, Picard RW (2013) MACH: my automated conversation coach. In: UbiComp, pp 697–706
3. Inoue K, Milhorat P, Lala D, Zhao T, Kawahara T (2016) Talking with ERICA, an autonomous android. In: Sigdial, pp 212–215
4. Kawahara T (2018) Spoken dialogue system for a human-like conversational robot ERICA. In: IWSDS
5. Kawahara T, Yamaguchi T, Inoue K, Takanashi K, Ward NG (2016) Prediction and generation of backchannel form for attentive listening systems. In: Interspeech, pp 2890–2894

6. Kobori T, Nakano M, Nakamura T (2016) Small talk improves user impressions of interview dialogue systems. In: Sigdial, pp 370–380
7. Smith MJ, Ginger EJ, Wright K, Wright MA, Taylor JL, Humm LB, Olsen DE, Bell MD, Fleming MF (2014) Virtual reality job interview training in adults with autism spectrum disorder. J Autism Develop Disord 44(10):2450–2463
8. Su MH, Wu CH, Huang KY, Hong QB, Huang HH (2018) Follow-up question generation using pattern-based seq2seq with a small corpus for interview coaching. In: Interspeech, pp 1006–1010

Automatic Head-Nod Generation Using Utterance Text Considering Personality Traits

Ryo Ishii, Taichi Katayama, Ryuichiro Higashinaka, and Junji Tomita

Abstract We propose a model for generating head nods from an utterance text considering personality traits. We have been investigating the automatic generation of body motion, such as nodding, from utterance text in dialog agent systems. Human body motion varies greatly depending on personality. Therefore, it is important to appropriately generate body motion according to the personality of the dialog agent. To construct our model, we first compiled a Japanese corpus of 24 dialogues including utterance, nod information, and personality traits (Big Five) of participants. Our nod-generation model also estimates the presence, frequency, and depth during each phrase by using various types of language information extracted from utterance text and personality traits. We evaluated how well the model can generate and estimate nods based on individual personality traits. The results indicate that our model using language information and personality trails outperformed a model using only language information.

1 Introduction

In human communication, non-verbal behavior is important to convey emotion and intention in addition to spoken language [3]. Therefore, in a dialogue system using conversational agents and robots, it is preferable to express appropriate nonverbal behavior according to the utterance and carry out smooth communication with a user.

Nodding is known to not only provide feedback to the partner but also affirmation, emphasis, turn-taking, rhythm, and utterance intention [9, 18, 21]. Also, nodding accompanying utterances strengthens the persuasive power of an utterance, making it easier for the other party to understand the content of the utterance [17]. Therefore, enabling a humanoid agent to nod when in dialogue with a human not only improves the impression of naturalness but also promotes conversation.

R. Ishii (✉) · T. Katayama · R. Higashinaka · J. Tomita
NTT Media Intelligence Laboratory, NTT Corporation, 1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa, Japan
e-mail: ishii.ryo@lab.ntt.co.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_26

Against such background, several studies have been conducted to generate nods during utterances, especially using speech-sound information such as prosody [2, 4, 6, 9, 14, 15, 23]. However, it was difficult to accurately generate nods from such information. The co-occurrence relation between speech-sound information and nodding in Japanese is known to be weak [8, 23].

We have been investigating the automatic generation of body motions, such as nodding, from utterance text in dialog agent systems [10–13]. For this study, we developed a model for generating detailed head nods for each phrase and estimating the nodding depth (the difference between the direction angle of the head at the beginning of nodding and the direction angle when the head is oriented furthest downward), presence, and frequency.

Human body motion varies greatly depending on the personality of the individual. Therefore, it is important to appropriately generate body motion according to the personality of the dialog agent. We believe that this allows users to feel comfortable and form better relationships with agents.

To construct our model, we first compiled a Japanese corpus of 24 dialogues including utterance, nod information, and personality traits (Big Five: openness to experience, conscientiousness, extraversion, agreeableness, and neuroticism) of participants. Our model also estimates nod presence, frequency, and depth during each phrase by using various types of language information extracted from utterance text and personality traits. We evaluated how well the model can generate nods reflecting the personality of individuals. The results indicate that our model using language information and personality traits outperformed a model using only language information.

2 Corpus

To create a Japanese conversation corpus including verbal and non-verbal behaviors for generating nods in dialogue, we recorded 24 face-to-face two-person conversations (12 groups of 2 different people). The participants were Japanese males and females in their 20s to 50s who had never met before. They sat facing each other (Fig. 1). To gather more data on nodding accompanying utterances, we adopted the explanation of a famous cartoon that participants had not seen as the conversational content. Before the dialogue, they watched a cartoon called "'Tom & Jerry" in which the characters do not speak. In each dialogue, one participant explained the content of the cartoon to the conversational partner within ten minutes. At any time during this period, the partner could freely ask questions about the content.

We recorded the participants' voices with a pin microphone attached to the chest and video recorded the entire discussion. We also took bust (chest, shoulders, and head) shots of each participant (recorded at 30 Hz). In each dialogue, the data on the utterances and nodding behaviors of the person explaining the cartoon were collected during the first half of the ten-minute period (120 min in total) as follows.

Fig. 1 Photograph of two participants having dialogue

- Utterances: We built an utterance unit using the inter-pausal unit (IPU) [16]. The utterance interval was manually extracted from the speech wave. A portion of an utterance followed by 200 ms of silence was used as the unit of one utterance. We collected 2965 IPUs. We also used J-tag [5], which is a general morphological analysis tool for Japanese, to divide an IPU into phrases. We collected a total of 11877 phrases.
- Head nod: A head nod is a gesture in which the head is tilted in alternating up and down arcs along the sagittal plane. A skilled annotator annotated the nods by using bust/head and overhead views in each frame of the videos. We regarded continuous nodding within a certain period as one nod event. The frequency (number) of nods was also manually labeled as 1, 2, 3, 4, or 5 or more. The change in the rotation

angle of the head when nodding occurred was measured using OpenFace, which is head-tracking software that uses image processing [1]. The difference between the direction angle of the head at the beginning of nodding and that when the head is oriented furthest downward was obtained as nodding-depth information. Nodding depth was classified into the following four stages.

- Micro: Depth less than 5 degrees
- Small: Depth greater than 5 degrees and less than or equal to 10 degrees
- Medium: Depth greater than 10 degrees and less than or equal to 20 degrees
- Large: Depth greater than 20 degrees

- Personality traits: We used the Big Five personality traits of each participant.

All verbal and non-verbal behavior data were integrated at 30 Hz for display using the ELAN viewer [22]. This viewer enabled us to annotate the multimodal data frame-by-frame and observe the data intuitively. In this study, we only handled utterance and head-nod data in the corpus we constructed. Nods occurred in 11877 out of the 5525 IPUs.

3 Head-Nod-Generation Model

We constructed our head-nod-generation model in three stages for each phrase using language information and personality traits. The presence or absence of nodding is first estimated. Next, when there is nodding, information on the frequency and depth is estimated independently. Finally, nodding is generated from these three estimation results.

3.1 Nodding Presence or Absence

First, we evaluated our model in terms of estimating the nodding presence or absence of participants using personality traits in addition to language information. We used the value of 5-dimensional personality characteristics of each participant as the feature values for generating head nods. We also use the following language information.

- Length of phrase (LP): Number of characters in a phrase.
- Word position (WP): Word position in a sentence.
- Bag of words (BW): Other studies focused on limited words to generate head nods. To handle more generic word information as well, we examined the number of occurrences of all words, not some words. We used J-tag [5], a general morphological analysis tool for Japanese.
- Dialogue act (DA): A DA was extracted using an estimation technique for Japanese [7, 19]. The technique can extract a DA using the word N-grams, semantic

Table 1 Evaluation results of our model for estimating nodding presence or absence

Feature values	Precision	Recall	F-score
Chance level	0.500	0.500	0.500
Language information	0.578	0.599	0.590
Language information+Personality traits	0.636	0.638	0.637

categories (obtained from a Japanese thesaurus Goi-Taikei), and character N-grams. There are 33 types of DAs.

- Part of speech (PS): Number of occurrences of the PSs of words in a phrase. We used J-tag [5] to extract PS information.
- Large-scale Japanese thesaurus (LT): The LT is a large lexical database of Japanese. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations.

Language information obtained from the target phrase and the phrases before and after the target phrase was taken as the feature value for the model.

We constructed our nod-generation model by using C4.5 [20], which is a well-known algorithm used to generate a decision tree, and evaluated its accuracy and the effectiveness of each type of language information. The criterion variable was a binary value as to whether a nod occurred.

We used 24-fold cross validation using a leave-one-person-out technique with the data from the 24 participants. To align the number of phrases of two classes, with or without nodding, the number of phrases in the larger class was reduced to be equal that in the smaller class. The total was 11050 phrases including 5525 phrases in which nodding occurred and 5525 phrases in which nodding did not occur. Therefore, chance level was 0.500.

We evaluated how well presence or absence of participants' nods could be estimated with an estimator generated from only data of other people. As shown in Table 1, the performance of our model in estimating the presence or absence of participants' nods using both language information and personality traits was higher than the model only using language information (the results of t-test were $p < 0.05$).

3.2 Estimating Nodding Frequency

We then evaluated our model in terms of estimating nodding frequency of participants using various types of language information. The objective values were categorized as 1, 2, 3, 4, or 5 and more. The feature values and machine-learning algorithm were

Table 2 Evaluation results of our model for estimating nodding frequency

Feature values	Precision	Recall	F-score
Chance level	0.144	0.380	0.209
Language information	0.348	0.370	0.339
Language information+Personality traits	0.396	0.378	0.387

Table 3 Evaluation results of our model for estimating nodding depth

Feature values	Precision	Recall	F-score
Chance level	0.273	0.299	0.282
Language information	0.381	0.420	0.397
Language information+Personality traits	0.447	0.412	0.430

the same as in the above case, and we determined which explanatory variable was valid.

We used 24-fold cross validation using a leave-one-person-out technique with the data for the 24 participants. The number of phrases was 5525 in which nodding occurred. We evaluated how well the nodding frequency of a participant could be estimated with an estimator generated only from data of other participants.

As shown in Table 2, the performance of our model using both language information and personality traits was higher than the model using only language information for estimating the nodding frequency of users (the results of t-test were $p < 0.05$).

3.3 Nodding Depth

Finally, we evaluated our model in terms of estimating nodding depth of participants using language information. The objective value was the numeric value of nodding depth. The nodding frequency estimated with our model was used as a feature value in addition to language information. The machine-learning algorithm was the same as in the above cases, and we determined which explanatory variable was valid.

We used 24-fold cross validation using a leave-one-person-out technique with the data for the 24 participants. The number of phrases was 5525 in which nodding occurred. We evaluated how well the nodding depth a participant could be estimated with an estimator generated only from data of other participants.

As shown in Table 3, the performance of our model using both language information and personality traits was higher than the model using only language information for estimating nodding depth of participants (the results of t-test were $p < 0.05$).

4 Discussion

The experimental results from our nod-generation model indicate that using the Big Five personality traits is useful for generating nods.

We used language information extracted from a unit of a phrase and before and after it and attempted to determine whether nodding occurred in the phrase. We did not consider the time-sequential information as a feature. In the future, we plan to construct a generation model with time-sequential information [12, 13].

We plan to evaluate how users perceive personality difference by observing the nods of a conversational agent generated with our model.

5 Conclusion

We proposed a head-nod-generation model that also estimates nodding presence, frequency, and depth during a phrase by using personality traits (Big Five) in addition to language information. The results indicate that our model using personality trait in addition to language information outperformed a model using only language information. The results also indicate that using personality traits is effective for estimating head nods of individuals.

In the future, we will focus on time-sequential language information to generate nods. We plan to construct a model for generating other body movements by considering personality traits. We have plans to evaluate how users perceive personality difference by observing agent nods generated with our model.

References

1. Amos B, Ludwiczuk B, Satyanarayanan M (2016) Openface: a general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science
2. Beskow J, Granstrom B, House D (2006) Visual correlates to prominence in several expressive modes. In: INTERSPEECH
3. BirdWhistell RL (1970) Kinesics and context. University of Pennsylvania Press
4. Busso C, Deng Z, Grimm M, Neumann U, Narayanan S (2007) Rigid head motion in expressive speech animation: analysis and synthesis. In: IEEE transactions on audio, speech, and language processing, pp 1075–1086
5. Fuchi T, Takagi S (1998) Japanese morphological analyzer using word cooccurrence -jtag. In: International conference on computational linguistics, pp 409–413
6. Graf HP, Cosatto E, Strom V, Huang FJ (2002) Visual prosody: facial movements accompanying speech. In: IEEE international conference on automatic face and gesture recognition, pp 381–386
7. Higashinaka R, Imamura K, Meguro T, Miyazaki C, Kobayashi N, Sugiyama H, Hirano T, Makino T, Matsuo Y (2014) Towards an open-domain conversational system fully based on natural language processing. In: International conference on computational linguistics, pp 928–939

8. Ishi CT, Haas J, Wilbers FP, Ishiguro H, Hagita N (2007) Analysis of head motions and speech, and head motion control in an android. In: IEEE/RSJ international conference on intelligent robots and systems, pp 548–553
9. Ishi CT, Ishiguro H, Hagita N (2010) Head motion during dialogue speech and nod timing control in humanoid robots. In: ACM/IEEE international conference on human-robot interaction, pp 293–300
10. Ishii R, Katayama T, Higashinaka R, Tomita J (2018) Automatic generation of head nods using utterance texts. In: 2018 27th IEEE international symposium on robot and human interactive communication (RO-MAN), pp 1143–1149
11. Ishii R, Higashinaka R, Nishida K, Katayama T, Kobayashi N, Tomita J (2018) Automatically generating head nods with linguistic information. In: Meiselwitz G (ed) Social computing and social media. Springer International Publishing, Cham, Technologies and analytics, pp 383–391
12. Ishii R, Katayama T, Higashinaka R, Tomita J (2018) Automatic generation system of virtual agent's motion using natural language. In: Proceedings of the 18th international conference on intelligent virtual agents, IVA '18, New York, NY, USA, 2018. ACM, pp 357–358
13. Ishii R, Katayama T, Higashinaka R, Tomita J (2018) Generating body motions using spoken language in dialogue. In: Intelligent virtual agents (IVA'18)
14. Iwano Y, Kageyama S, Morikawa E, Nakazato S, Shirai K (1996) Analysis of head movements and its role in spoken dialogue. In: International conference on spoken language, pp 2167–2170
15. Munhall KG, Jones JA, Callan DE, Kuratate T, Vatikiotis-Bateson E (2004) Visual prosody and speech intelligibility: head movement improves auditory speech perception 15(2):133–137
16. Koiso H, Horiuchi Y, Tutiya S, Ichikawa A, Den Y (1998) An analysis of turn-taking and backchannels based on prosodic and syntactic features in japanese map task dialogs. Lang Speech 41:295–321
17. Lohse M, Rothuis R, Gallego-Pérez J, Karreman DE, Evers V (2014) Robot gestures make difficult tasks easier: the impact of gestures on perceived workload and task performance. In: Proceedings of the SIGCHI conference on human factors in computing systems, CHI '14, New York, NY, USA, 2014. ACM, pp 1459–1466
18. McBreen HM, Jack MA (2001) Evaluating humanoid synthetic agents in e-retail applications. IEEE Trans Syst, Man, Cybern - Part A: Syst Humans 31:5
19. Meguro T, Higashinaka R, Minami Y, Dohsaka K (2010) Controlling listening-oriented dialogue using partially observable markov decision processes. In: International conference on computational linguistics, pp 761–769
20. Quinlan JR (1996) Improved use of continuous attributes in c4.5. J Artif Intell Res 4:77–90
21. Watanabe T, Danbara R, Okubo M (2003) Effects of a speech-driven embodied interactive actor interactor on talker's speech characteristics. In: IEEE international workshop on robot-human interactive communication, pp 211–216
22. Wittenburg P, Brugman H, Russel A, Klassmann A, Sloetjes H (2006) Elan a professional framework for multimodality research. In: International conference on language resources and evaluation
23. Yehia HC, Kuratate T, Vatikiotis-Bateson E (2002) Linking facial animation, head motion and speech acoustics 30(3):555–568

Opinion Building Based on the Argumentative Dialogue System BEA

Annalena Aicher, Niklas Rach, Wolfgang Minker, and Stefan Ultes

Abstract In this work, we introduce BEA, an argumentative Dialogue System that assists the user in his or her opinion forming regarding a certain controversial topic. To this end, we establish an opinion model based on weighted bipolar argumentation graphs that allows the system to infer the influence of preferences expressed by the user on all related aspects and is updated by the system in real time during the interaction. The system and the model are tested and discussed by use of an argument structure consisting of 72 components in a proof of principal scenario, showing a high sensitivity of the employed model regarding the expressed preferences.

1 Introduction

Over the past 20 years the amount of available data has grown considerably. With the overwhelming and often contradicting information present on the internet, it is quite difficult to retrieve and extract meaningful, accurate and suitable information. Thus, advanced tools and systems that enable the user to find the adequate information and make choices that meet their needs and expectations are required. This is also reflected in the recent development of recommender and decision support systems that have become increasingly popular in assisting users with their choices [21]. To enable such assistance systems to offer information that is tailored to a particular user, it is essential to understand the user and his or her individual interests, opinions, priorities and thus, preferences. Especially in situations where hands-free interaction is required (like in the context of smart home or smart environment), the need for

A. Aicher (✉) · N. Rach · W. Minker · S. Ultes
Institute of Communication Engineering, Ulm University, Ulm, Germany
e-mail: annalena.aicher@uni-ulm.de

N. Rach
e-mail: niklas.rach@uni-ulm.de

W. Minker
e-mail: wolfgang.minker@uni-ulm.de

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_27

mediating agents that are able to process natural language requests and master the complicated structure of the network is given [10]. Recent work in Natural Language Processing (NLP), e.g. in the field of Argument Mining [11, 13, 14], is concerned with the automatic extraction and mining of content of the Internet. However, this data structure is non-intuitive for humans and requires an interface to be accessed by the user.

In this work we introduce the Argumentative Dialogue Systems *BEA* ('build evidence-based argumentation') as such an interface which provides an incremental and intuitive access to available information [16] on a certain topic. The system utilizes data structured according to the argument mining scheme introduced in [23] as database and presents related aspects (arguments) incrementally during the interaction. In order to establish a complex and fine grained model of the users opinion, we realize the user preference model based on bipolar weighted argumentation graphs (WBAGs) [3] sketched in [17]). This model allows the user to introduce preferences between arguments related to the same sub-topic as well as to automatically infer the effect of this preference on the overall stance. Hence, the system is able to model the personal opinion of the user based on the expressed preferences, resulting in a opinion model which considers the available information as well as the users personal views.

The remainder of the paper is as follows: Sect. 2 examines related work on argumentative Dialogue Systems and user preferences in recommender and decision support systems. Section 3 introduces the employed preference model which maps the opinion of the user. In Sect. 4, we introduce the architecture of the system in combination with the theoretical background of the respective components. Subsequently, in Sect. 5, the results are discussed exemplary. We close with a conclusion and a brief discussion of future work in Sect. 6.

2 Related Work

In the following we give an overview on related work on decision support and recommender systems, as well as Dialogue Systems based on argumentation graphs. Recent examples of the latter where introduced in [18, 20, 22] but focus on a persuasive setup, i.e. the exchange of arguments with the user. In contrast, we model a cooperative user-system interaction with a focus on the user's preference and opinion regarding the available arguments in order to extend the underlying argumentative technology to assistive scenarios.

The need for bipolar argumentation graphs has been empirically supported by [15]. In recent years, some *weighted bipolar argumentation frameworks* have been introduced that start with an initial weight of arguments and adapt this weight based on the strength of attacking and supporting arguments [2, 3, 6, 9, 19]. For example, in the analysis of discussions in social networks, the initial weight can be based on the number of likes, dislikes and shares of a post, and attack and support relations can be identified from replies using sentiment analysis tools [1]. Another application is decision support in areas like engineering design and e-democracy [5, 19]. However, those systems are focused on quantitative decision support problems only and do not involve an immediate user interaction via dialogue. In contrast, the herein discussed approach includes a domain-independent opinion model that is developed during the interaction and based on the user's personal views.

Also, most decision making support systems [4, 21] as well as recommender systems [12] are clearly focused on the application concerning specific items, like e.g. movie recommendations [8]. An overview on the state of the art and methodology of recommender systems, as well as trends is given by Ricci et al. [24]. Thus, most existing research in this area is based purely on quantitative aspects such as indices of popularity or measures of similarity between items or users. Another approach is presented by Rago et al. [19]. It builds upon an item/aspect-based graph with users partially given ratings and maps them onto so-called Tripolar Argumentation Frameworks, experimentally evaluated on a movie dataset. In contrast, the herein presented approach is based on the active involvement of the user via dialogue, which enables us to directly access the user's preferences and opinions.

3 The User Preference Model

In this section, we introduce an opinion model based on bipolar argument structures which encodes dependencies of different arguments in one of two relations (*support* or *attack*) between nodes in a graph. As classical bipolar argumentation structures only capture the relations between arguments and do not distinguish their validity or persuasiveness, we utilize *weighted bipolar argumentation graphs* (BAGs) [2, 3] in which a *weight* is assigned to each argument. These weights encode the personal opinion of the user regarding the respective arguments and are updated during the interaction based on the expressed preferences. The overall *strength* of an argument can then be determined from the weight and the strength of its attackers and supporters, i.e. arguments that are related to it. For the sake of simplicity and without loss of generality, we choose weights and strengths of arguments to be a real valued number in the interval [0, 1]. Approaches for computing strength values also often assume an acyclic argumentation graph which is ensured in the herein discussed scenario by the data structure [16].

We utilize the Euler-based restricted semantics introduced by Amgoud and Ben-Naim [3], which aggregates the strength of arguments in a linear fashion. The energy E_i (also Euler's number) at an argument i is defined as [3]:

$$E_i = \sum_i s_{i,\text{sup}} - \sum_i s_{i,\text{att}} \tag{1}$$

where 'i,sup' and 'i,att' are the sets of supporters and attackers and s_i denotes the corresponding strength value. Hence, the stronger or more-numerous the supporting argument components are, the greater and more-likely-positive is that exponent (and

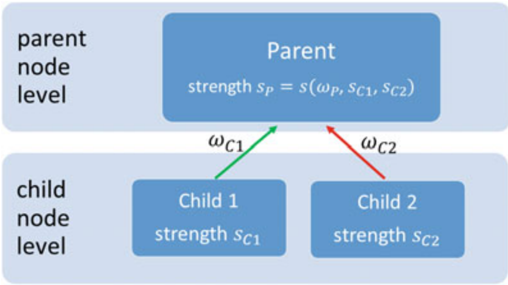


Fig. 1 Illustration of a weighted bipolar argumentation graph, including the strengths and weights

vice versa for attackers). The aggregated strength of an argument i is a function of its initial weight ω_i and the energy in (1) [3]:

$$s_i = 1 - \frac{1 - \omega_i^2}{1 + \omega_i e^{E_i}}. \tag{2}$$

Consequently, s_i considers the weight of the argument itself as well as the influence from connected arguments. If an argument has no child nodes, its strength equals its weight. Figure 1 shows an exemplary sketch of the discussed structure. A parent node is sketched with its supporting and rejecting child nodes. Each node has a specific weight and strength, which are defined further on. Our opinion model considers preferences between arguments related to the same parent node. After a preference is expressed, the corresponding weight is adjusted according to an update function and iterated through all connected arguments following Eq. 2. Preferences can be expressed as either *prefer* or *reject*, where the first option means that the argument is preferred over its siblings and the latter means that the argument is considered invalid. Consequently, the strength of a preferred argument has to be greater than the ones of its siblings and thus set to

$$s_i' = s_{\max} + 0.5\,(1 - s_{\max})\,. \tag{3}$$

Here, s_{max} denotes the maximum strength of all siblings. As we formally update only weights in order to consider later preferences, we determine the new weight (solution of Eq. 2) as

$$\omega_i' = \frac{e^{E_i}\left(1 - s_i'\right)}{2}\left(-1 \pm \sqrt{1 + \frac{4s_i'}{e^{2E_i}\left(1 - s_i'\right)^2}}\right). \tag{4}$$

Due to the square root, there are two solutions to this equation but only one meets this in the required interval [0, 1]. If an argument is rejected, its weight (and thus its strength as well) is set to 0. The whole opinion model is updated after each expressed preference according to the following scheme:

1. According to the user's preference (*prefer* or *reject*) regarding a certain argument i (node), its new strength s_i' is calculated.
2. The new energy of the parent node of i is determined by applying Eq. (1). Graphically illustrated we move up a level in the graph depicted in Fig. 3. Using the new energy of the parent of i calculated in the previous step and the update formula in Eq. (2), the new strength of the parent node is calculated.
3. Step three is repeated until all related values are updated.

The overall preference of the user can then be determined by calculating the energy of the root node. As no information about the user's preferences is known before the interaction, we initialize all weights with the same value ($\omega_0 = 0.5$, representing indifference).

4 BEA

In this section we discuss our argumentative Dialogue System BEA which serves as an interface between the user and the argument structure. The graphical surface of the system includes a drop-down menu from which the desired action can be chosen by the user, a dialogue history and the system response. A screen shot of an ongoing interaction with the system is shown in Fig. 2. Here, the orange box marks the drop down menu, where the user is able to choose between the possible moves. The system's output after the user chose his move is marked by the red box. To ensure a better traceability of the dialogue history, the whole dialogue is displayed on the right hand side (green box). We discuss the utilized data structure (or knowledge base), the interaction (or dialogue) model and the natural language generation (NLG) in detail in the following subsections.

4.1 Data and Argument Structure

In the following we take a look on the employed annotation scheme and the data it is applied to. The latter corresponds to the one presented in Rach et al. [16] consisting of a sample debate from the *Debatabase* of the idebate.org[1] website. As pointed out by Rach et al. one of its advantages are the secured quality standards regarding both form and content. Furthermore, the representation of pro and con side in all debates meets our purpose to offer the user a fair chance to decide unprejudiced which side to prefer. Moreover the specific structure provided by *Debatabase* debates allows

[1]https://idebate.org/debatabase (last accessed 09 January 2018). Material reproduced from www.iedebate.org with the permission of the International Debating Education Association.

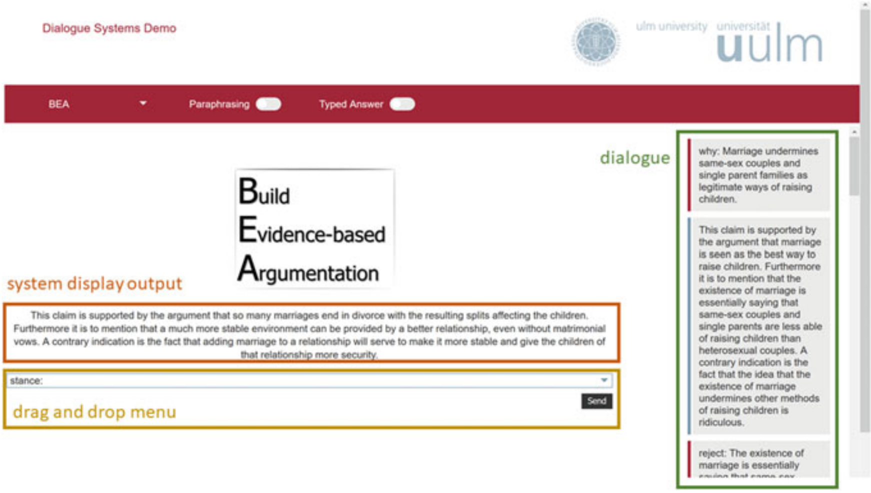


Fig. 2 Graphical surface of BEA, consisting of a drop down menu (orange), a system display output (red) and the dialogue history (green)

quick screening for suitable topics and facilitates the argument annotation process later on. As a sample debate for our work we chose the topic *Marriage is an outdated institution* due to its large number of arguments.

The utilized annotation scheme meets the requirements of the discussed argument-based preference model described in Sect. 3 and was introduced by Stab et al. [23] for annotating argumentative discourse structures and relations in persuasive essays. According to Stab et al. an argument consists of several components (*major claim*, *claim* and *premise*). Usually a single *major claim* formulates the overall topic of the debate, representing the root node in the graph (here *Marriage is an outdated institution*). Thus it is the only component without target. *Claims* on the other hand are allegations which formulate a certain opinion targeting the *major claim* but still need to be justified by further arguments, *premises* respectively. Hence, a claim is either supported or attacked by at least one other premise. We will further on only focus on non-cyclic graphs, meaning that each premise only targets one other component, leading to a strictly hierarchical structure. Furthermore the annotation scheme distinguishes two directed relations a premise can have towards a claim (*support* and *attack*). Between sibling nodes there exists no explicit relation.

In order to build the opinion model, we assign a weight ω_i to each argument in the database which is used to determine the current user preference. This is depicted in Fig. 3. The enumerated levels illustrate the depth of an argumentation line. The green arrows denote a supporting relation towards an argument component whereas the red ones mark an attack.

The enumeration of levels displays how deep an argumentation line reaches and will in Sect. 5 be important to analyze to which extent our employed model is sensitive to preference changes in higher levels and thus, very selective parts of the

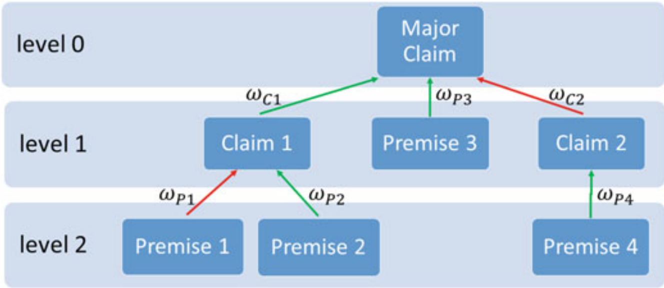


Fig. 3 Illustration of a structured graph representing the annotated argument components

argumentation line. Note that Fig. 3 shows an exemplary graph with three levels and the approach can be generalized to an arbitrary graph depth.

As described by Rach et al., the employed structure consists of a total of 72 argument components (1 *major claim*, 10 *claims* and 61 *premises*) and their corresponding relations and encoded in an OWL ontology [7] for further use. In order to generate the dialogue as natural as possible the original annotated sentences were slightly modified to create complete and reasonable utterances. Due to the generality of the annotation scheme, the system is not restricted to the herein considered data and generally every argument structures that can be mapped into the applied scheme can be processed by the system.

4.2 The Dialogue Model

In order to navigate through the often large and complex argument structure, we restrict the interaction on sub-structures consisting of one parent node or current *state* and all related child nodes. Thus, the user generally is able to express preferences between the child nodes of the current state or navigate to another one (and thus another sub-structure). The interaction between the user and system is separated in turns, where each turn consists of a user action and a system response. Each user action is as a request the system is processing, followed by a corresponding natural language answer. The system updates the opinion model and the current *state* according to the user's choice after the respective action. The response of the system depends on the available arguments provided by the argument tree discussed in Sect. 4.1 and the current opinion model. Thus, the user is enabled to explore available arguments step by step, to express preferences between them and to request information about the influence of his choice on the stance of the current state. We distinguish the six user actions *why*, *prefer*, *reject*, *stance*, *level up* and *exit*. Each move is explained in detail in Table 1, including it's influence on the state and the preference model. Each interaction starts with a greeting of the system and a repetition of the discussed topic. Afterwards, the resulting dialogue is determined

Table 1 Available moves and explanation

Moves	Explanation	Determiners
Stance	Stance move which returns the current stance of the user by calculating the energy of the corresponding node	Always possible
Exit	Termination move which immediately terminates the conversation with an automated farewell phrase	Always possible
Level up	Ascent move which changes the current state and switches to the according parent node (one level up)	Always possible. In case that the current node is already the root node this move leads to itself
Why (child node)	Information-seeking move which asks for argument components related to the chosen child node. Changes the state to the chosen node (*level down*)	Only possible if child nodes exist
Prefer	Preference move if a user supports an argument component over the other ones. Preferences can be stated between siblings, as we imply a direct relation. Strengths and weights are updated according to Sect. 3	Always possible
Reject	Reject move if a user abandons an argument component. Hence, also all its children are excluded from the graph and its energy is thus zero. Strengths and weights are updated according to Sect. 3	Always possible

only by the user and his choices. In each turn all moves, and if child nodes exist, also all premises with the according moves are shown and the user decides what to do next. The termination criterion for the dialogue is given by the user choosing the exit move.

4.3 Natural Language Generation

The NLG of the system is based on the original textual representation of the argument components. As discussed in Sect. 4.1, the annotated sentences were slightly modified to form a stand-alone utterance which serves as a template for the respective system response.

Additionally, a list of natural language representations for each type of move was defined. During the generation of the utterances, the explicit formulation and introductory phrase is chosen from this list randomly. Regarding the *why* move the relation (*support* or *attack*) of the presented children is clearly stressed by specific formulations, such as *This claim is supported by the argument that...* and *A contrary indication is the fact that...*. Especially if there exists more than one child node this generates a contextual connection, which forms a continuous utterance and makes the dialogue more natural. In order to signalize the user that his or her *prefer* or *reject* move has been taken into account, a notification sentence is generated, like *Further on I will take into account that you rejected/preferred the presented argument*. Likewise, we create notifications, such as (*I gladly present you the previous argument again.*), for the *level up* move. In case the user asks for his stance, the system returns the following phrases *At this moment you are in favor of/indifferent concerning/refuse this argument.* corresponding to the value of the energy of this node. Finally, a finishing phrase is shown if the *exit* move is chosen in order to mark that the dialogue is finished.

5 Discussion

In this section, we discuss exemplified extracts of the interaction between the system and the user. As the interaction takes places via a drop down menu, where the user is able to choose his actions and the system answers by monitor output, the user side is modelled without utterance output.

In the following, we demonstrate the functionality of the whole system at exemplary extracts of the user system interaction. This includes the interaction itself as well as the iterative update of the underlying opinion model.

In the first extract, we concentrate on the claim[2] "*Marriage undermines same-sex couples and single parent families as legitimate ways of raising children*", which is addressed by the user with a *why* move.

System: "*This claim is supported by the argument that marriage is seen as the best way to raise children (C1). Another argument in favor is the existence of marriage is essentially saying that same-sex couples and single parents are less able of raising children than heterosexual couples (P3). A contrary indication is the fact that the idea that the existence of marriage undermines other methods of raising children is ridiculous (C2).*"

User: reject C1

System: "*Further on I will take into account that you rejected the presented argument.*"

Figure 4 illustrates the corresponding update of the opinion model after the reject of C1. As expressed by the grey color, by rejecting C1, the two premises P1 and P2 become irrelevant. Thus, in the subsequent conversation this branch is excluded. In the next extract the dialogue is continued and the user asks for further information on C2 (*why* move).

[2]For the sake of simplicity we define this argument as the Major Claim of this subdialogue.

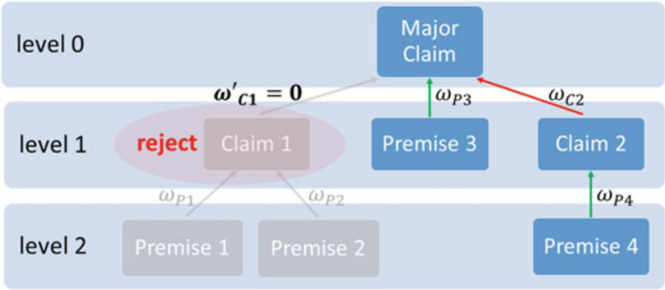


Fig. 4 Illustration of the *reject* move. The new weight of C1 ω'_{C1} is set to zero. All corresponding child nodes are no longer relevant, which is expressed in the grey masking of P1 and P2. The resulting graph consists of three argument components, which are left (P3, C2 and P4)

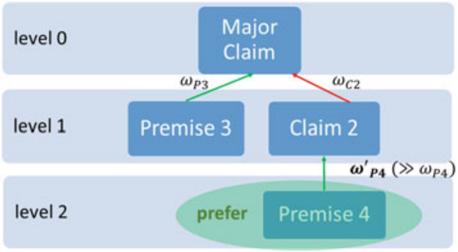


Fig. 5 Illustration of the *prefer* move. The graph consists of the two remaining branches. The new weight of P4 ω'_{P4} is significantly bigger than old the one

System: "*This claim is supported by the argument that some people choosing to raise children in a certain way does not prevent or inhibit other people doing so in a different way (P4).*"
User: prefer P4
System: "*Further on I will take into account your preference.*"
User: stance
System: "*At this moment you are in favor of C2.*"
User: level up
System: "*Would you like to hear the previous argument?*"
User: stance
System: "*At this moment you refuse this argument (Major Claim).*"

As shown in Fig. 5 the *prefer* move does not change the graph like the *reject* move did before, but again the weights and strengths are updated. This can clearly be seen regarding the system response to the *stance* moves. In case of P4 the user stated a preference and thus, the system recognizes that the user is in favor of C2 as it is *supported* by P4. As C2 *attacks* the Major Claim, consequently the user refuses the latter.

To test the consistency and sensitivity of the system, different combinations of *prefer* and *reject* on all levels of the argument structure were tested. It could be show, that even a preference/rejection choice on a high level can influences the stance of the

Major Claim. Hence, we value the herein presented preference-based model and the bipolar weighted argumentation as an adequate approach to reflect user preferences and furthermore, build an opinion on controversial topics.

6 Conclusion and Outlook

This work addressed the machine aided opinion forming and preference modelling by use of argumentative Dialogue Systems. We have discussed and implemented a fine grained opinion model based on BAGs which allows to infer the influence of expressed user preferences on related aspects of the topic. Moreover, we introduced the the argumentative Dialogue System BEA which allows users to explore all aspects of a certain topic and to express preferences between them. A proof of principle investigation showed that the users choices on any level influence the overall stance (and thus all related arguments as well) and that the model is thus sensitive enough for the desired purpose.

In future work we aim for an extended user survey to evaluate the informative value of the opinion model. Also a more detailed comparison with regard to related approaches will extend the evaluation of our proposed model. Furthermore, we will extend BEA to enable an interaction with the system via speech. Moreover, we will compare alternatives for the updating of the graph and the default initialization by using further information on the user, e.g. preferences in other discussions.

Acknowledgements This work has been funded by the Deutsche Forschungsgemeinschaft (DFG) within the project "How to Win Arguments - Empowering Virtual Agents to Improve their Persuasiveness", Grant Number 376696351, as part of the Priority Program "Robust Argumentation Machines (RATIO)" (SPP-1999).

References

1. Alsinet T, Argelich J, Béjar R, Fernández C, Mateu C, Planes J (2017) Weighted argumentation for analysis of discussions in twitter. Int J Approx Reason 85
2. Amgoud L, Ben-Naim J (2016) Evaluation of arguments from support relations: axioms and semantics. In: Proceedings of the twenty-fifth international joint conference on artificial intelligence, IJCAI-16, pp 900–906. International Joint Conferences on Artificial Intelligence Organization (2016)
3. Amgoud L, Ben-Naim J (2018) Weighted bipolar argumentation graphs: axioms and semantics. In: Proceedings of the twenty-seventh international joint conference on artificial intelligence, IJCAI-18, pp 5194–5198. International Joint Conferences on Artificial Intelligence Organization
4. Amgoud L, Prade H (2009) Using arguments for making and explaining decisions. Artif Intell 173
5. Baroni P, Romano M, Toni F, Aurisicchio M, Bertanza G (2013) An argumentation-based approach for automatic evaluation of design debates. Computational logic in multi-agent systems. Springer, Berlin, pp 340–356

6. Baroni P, Romano M, Toni F, Aurisicchio M, Bertanza G (2015) Automatic evaluation of design alternatives with quantitative argumentation. Argument Comput 6(1):24–49
7. Bechhofer S (2009) Owl: web ontology language. Encyclopedia of database systems. Springer, Berlin, pp 2008–2009
8. Briguez C, Budán M, Deagustini C, Maguitman A, Capobianco M, Simari G (2014) Argument-based mixed recommenders and their application to movie suggestion. Expert Syst Appl 41(14):6467–6482
9. Budán M, Simari G, Simari G (2016) Using argument features to improve the argumentation process. In: Computational models of argument - proceedings of COMMA 2016, Potsdam, Germany, 12–16 September, 2016, pp 151–158
10. Jokinen K (2018) Natural language and dialogue interfaces
11. Lippi M, Torroni P (2016) Argumentation mining: State of the art and emerging trends. ACM Trans Int Technol (TOIT) 16(2):10
12. Lops P, de Gemmis M, Semeraro G (2011) Content-based recommender systems: state of the art and trends. Springer, Boston, pp 73–105
13. Moens MF (2013) Argumentation mining: Where are we now, where do we want to be and how do we get there? In: Post-proceedings of the 4th and 5th workshops of the forum for information retrieval evaluation, p 2. ACM
14. Palau RM, Moens MF (2009) Argumentation mining: the detection, classification and structure of arguments in text. In: Proceedings of the 12th international conference on artificial intelligence and law, pp 98–107. ACM
15. Polberg S, Hunter A (2018) Empirical evaluation of abstract argumentation: supporting the need for bipolar and probabilistic approaches. Int J Approx Reason 93:487–543
16. Rach N, Langhammer S, Minker W, Ultes S (2018) Utilizing argument mining techniques for argumentative dialogue systems. In: Proceedings of the 9th international workshop on spoken dialogue systems (IWSDS)
17. Rach N, Weber K, Aicher A, André E, Minker W (2018) Emotion recognition based preference modelling in argumentative dialogue system. Submitted to the 1st international workshop on pervasive computing and spoken dialogue systems technology (PerDial)
18. Rach N, Weber K, Pragst L, André E, Minker W, Ultes S (2018) Eva: a multimodal argumentative dialogue system. In: Proceedings of the 2018 on international conference on multimodal interaction, pp 551–552. ACM
19. Rago A, Toni F, Aurisicchio M, Baroni P (2016) Discontinuity-free decision support with quantitative argumentation debates. In: Fifteenth international conference on principles of knowledge representation and reasoning (KR 2016), pp 63–73
20. Rakshit G, Bowden K, Reed L, Misra A, Walker M (2017) Debbie, the debate bot of the future. arXiv preprint arXiv:1709.03167
21. Ricci F, Rokach L, Shapira B (2015) Recommender systems: introduction and challenges. In: Ricci F, Rokach L, Shapira B (eds) Recommender systems handbook. Springer, Boston
22. Rosenfeld A, Kraus S (2016) Strategical argumentative agent for human persuasion. In: ECAI, pp 320–328
23. Stab C, Gurevych I (2014) Annotating argument components and relations in persuasive essays. In: COLING, pp 1501–1510
24. Tuzhilin A, Adomavicius G (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17:734–749

Lifelong Learning

Learning Between the Lines: Interactive Learning Modules Within Corpus Design

Maria Di Maro, Antonio Origlia, and Francesco Cutugno

Abstract The present paper reports on the advantages of learning inferences and understanding strategies from the interactive structure of a corpus. First of all, we introduce the SUGAR corpus for the cooking domain, describing its peculiar collection and annotation procedures. After this first overview, we show how information included within the corpus can be used to enhance the action interpretation in dialogue systems. This can be the case of linguistic elements or related lexical units which can be acquired from a linked database or from rephrasing strategies within the corpus itself. In all the AI-based approaches depending on a training process using large and representative corpora, the probability to correctly predict the creativity a speaker can perform in using language is lower than expected. Trying to capture most of the possible words and expressions a speaker could use is extremely necessary, but even an empirical, finite collection of cases could not be enough. For this reason, the use of our corpus, possibly in combination with online training, appears as an appealing solution.

M. Di Maro (✉)
Department of Humanities, Università degli Studi di Napoli 'Federico II', Napoli, Italy
e-mail: maria.dimaro2@unina.it

A. Origlia · F. Cutugno
Department of Electrical Engineering and Information Technology,
Università degli Studi di Napoli 'Federico II', Napoli, Italy
e-mail: antonio.origlia@unina.it

F. Cutugno
e-mail: cutugno@unina.it

A. Origlia
URBAN/ECO Research Center, Napoli, Italy

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_28

1 Introduction

Spoken dialogue systems are nowadays in the spotlight in different commercial, academic and industrial sectors. The data-driven approach is becoming more and more important to infer knowledge and communicative strategies in the field of spoken language understanding and generation for applying different statistic and machine learning algorithms in solving problems [14]. The corpora available for this purposes are diverse, depending on the specific research goals: spoken, spontaneous and human-human, such as the Switchboard dataset [4]; spoken, specialised and human-machine, such as the Let's Go! corpus [11]; written, spontaneous and human-human, such as the NPS Chat Corpus [3]; spoken, general-purpose and human-human, such as the OpenSubtitles collection [15].

Despite the great amount of data which is available to be exploited for the language design of such systems, corpora can still be perceived as limited in terms of language usages which are not described nor implied within it. For this reason, different techniques are being applied to let conversational agents learn from the interaction itself. Among the others, we mention the reinforcement learning techniques, based on live interactions with humans (online) or user simulator models (offline), which can make use of partially observable Markovian Decision Processes (POMDP) for the selection of the correct action according to the interlocutor feedback [10, 18]. In this scenario, different scholars have pointed out the importance of learning while interacting, (i) by asking questions as students would do for the improvement of the learning process [7]; (ii) by exploiting users feedback online through imitation and rewards analysis to enlarge the fixed dialogue corpora-based offline learning [8]; (iii) by using corpora to train a reinforcement learning system and to bootstrap clarification strategies [5, 6, 12, 13].

As a matter of fact, despite their limitations, corpora are still the most exploited source for learning purposes. Therefore, in this paper, we want to show how the clarification strategies could be sometimes avoided during the actual interaction by learning inferences based on the rephrasing of commands by users, following clarification requests uttered by the system during the corpus collection, as being a robust basis for a lifelong learning system. In particular, we are going to present our SUGAR (Spoken Utterances Guiding Chef's Assistant Robots) Corpus [2] whose peculiar interactive learning structure can be used for the interpretation of unclear commands, in situation where clarification strategies would have been applied. Moreover, the active learnability of the corpus-based knowledge can be initiated in combination with a domain-specific database that can assure a wide range of applicability in different situations. The paper is structured as follows: in the next section (Sect. 2) we are going to present our corpus and its interactive character lying in its collection procedure; afterwards, in Sect. 3, we are going to describe how its interactive nature can be exploited for learning from the rephrasing of unclear commands and how its arguments-filling structure can make it suitable for ontology-based learning purposes.

2 Corpus Description

The SUGAR corpus[1] was created for EVALITA 2018 [1, 2] and contains 2293 audio files corresponding to Italian cooking actions annotated through predicate-arguments structures [16]. To collect the corpus, a 3D virtual environment was designed. We designed a real kitchen in Unreal Engine 4,[2] which could be virtually visited by means of the Oculus Rift.[3] In this kitchen users could interact with a robot - named Bastian - which received commands to accomplish some recipes, guided via a Wizard-of-Oz. User's orders were triggered by silent cooking videos shown in a TV screen put in the 3D scene, thus ensuring the naturalness of the spoken production. Videos were segmented into elementary portions and sequentially proposed to the speakers who uttered a single sentence after having seen each single frame (Fig. 1). The collected corpus thus consists of a set of spoken commands, whose meaning derives from the various combination of actions, items (i.e. ingredients), tools and different modifiers.

Actions are represented as a finite set of generic predicates accepting an open set of domain-dependent parameters, as follows

$$put(pot, fire)$$

The annotation process resulted in determining the optimal predicate-argument structure corresponding to each command, according to the action templates previously defined through the selected video collection[4] (Table 1). In the annotation files, the symbols are univocal: square brackets are used to indicate a list of ingredients, slashes indicate the alternative among possible arguments, asterisks are used when an argument is not explicitly instantiated but recoverable from the context (i.e. previous instantiated arguments, which are not uttered, not even by means of clitics or other pronouns) or from the semantics of the verb (i.e. instrumental verbs). For instance, *fry(flowers)* is represented as *add(flowers, *oil*)* because *oil* is implicitly expressed in the semantics of the verb *to fry* as an instrument to accomplish the action. Among other phenomena, it is worth mentioning the presence of actions paired with templates, even when the syntactic structure needed a reconstruction, as in *cover(bowl, wrap)* which is annotated with a more generic template as *put(wrap, bowl)*. In other cases, the uttered action represents the consequence of the action reported in the template, as in *separate(part, flowers)* annotated as *clean(flowers)*, or *stir([yeast, water])* represented with *melt(yeast, water)*. The arguments order does not reflect the one uttered in the recorded audio files, but the following:

$$action(quantity, object, complement, modality)$$

[1] Available on GitHub: https://github.com/evalitaunina/SUGAR_Corpus.

[2] www.unrealengine.com.

[3] www.oculus.com.

[4] The videos were selected from the *Giallo Zafferano* website: https://www.giallozafferano.it/.

Fig. 1 3D Reconstruction of Bastian in his Kitchen. On the wall, the television showing frames of video recipes, from which users could extract actions to utter as commands

Table 1 Action templates

Predicate	Arguments
prendere *take*	quantità, [ingredienti]/recipiente *quantity, [ingredients]/container*
aprire *open*	quantità, [ingredienti], recipiente *quantity, [ingredients], container*
mettere *put*	quantità, utensile/[ingredienti], elettrodomestico, modalità *quantity, tool/[ingredients], appliance, modality*
sbucciare *peel*	quantità, [ingredienti], utensile *quantity, [ingredients], tool*
schiacciare *mash*	[ingredienti], utensile *[ingredients], tool*
passare *pass through/strain*	[ingredienti], utensile *[ingredients], tool*
grattare *shred*	[ingredienti], utensile *[ingredients], tool*
girare *turn*	[ingredienti], utensile *[ingredients], tool*
togliere *remove*	utensile/prodotto, elettrodomestico *tool/product, appliance*
aggiungere *add*	quantità, [ingredienti], utensile/recipiente/ elettrodomestico/[ingredienti], modalità *quantity, [ingredients], tool/container/ appliance/[ingredients], modality*
mescolare *stir*	[ingredienti], utensile, modalità *[ingredients], tool, modality*
impastare *knead*	[ingredienti] *[ingredients]*
separare *split*	parte/[ingredienti], ingrediente/utensile *part/[ingredients], ingredient/tool*
coprire *cover*	recipiente/[ingredienti], strumento *container/[ingredients], instrument*
scoprire *uncover*	recipiente/[ingredienti] *container/[ingredients]*
controllare *check*	temperature, ingrediente *temperature, ingredient*
cuocere *cook*	quantità, [ingredienti], utensile, modalità *quantity, [ingredients], tool, modality*

The modality[5] arguments are diverse and follow a specific order: *adverb*, *cooking modality*, *temperature* and *time*.

Although the action templates are limited in number, they can represent different types of action, as shown in the previous examples. Moreover, despite the small size of the corpus, it can be used as a training set for learning purposes, considering the sectoral openness of the arguments slots (Sect. 3).

3 Interactive Learning Modules

The purpose of this paper is to show how the interactive learning advantages could be exploited starting from the corpus collection and structure. For this purpose, we talk here about *clarification-based learning* and *ontology-based learning*. With

[5]The quantity always precedes the noun it is referred to. Therefore, it can also occur before the complement.

the former, we refer to the possibility of deriving knowledge and skills from the linguistic analysis of linked rephrased utterances (i.e. implicitness of arguments, anaphora resolution, semantic relationships recognition, etc), whereas with the latter we mean that the system can actively learn from an external database enabling the automatic lexical extension of the corpus and thus the understanding of potentially any possible recipe. Through our learning modules we, therefore, intend to present two possible corpus collection structures which can be used within a functioning systems, making use of either grammars or machine learning algorithms.

3.1 Clarification-Based Learning

As far as the clarification-based learning is concerned, during the collection phase, our Bastian could ask for clarifications to users in two different scenarios: (i) when the action was not correspondent to the one shown in the television (*I think this is not the correct action*); (ii) when the command was not clear, because of the lack of a not-previously mentioned argument or of a wrongly selected argument or action (*I don't understand. Could you repeat?*). This second scenario is particularly useful to extract the adequate knowledge to train the system to connect correct utterances to less precise ones. For training purposes, in addition to what has been done in the annotation process for EVALITA 2018, we enriched the corpus with the less clear instances (left out in the EVALITA task) which were paired with the more precise ones. The couples were, furthermore, annotated as far as the occurring language phenomenon was concerned. In the following examples, we exemplify some recurrent inaccuracies with their corresponding corrections.

(1) a. ***[ ]*** *due uova.*
 '[ ] two eggs'.
 b. ***Rompi*** *due uova.*
 'Crack two eggs'.

(2) a. *Aggiungi le uova* ***[ ].***
 'Add the eggs [ ]'.
 b. *Aggiungi le uova* ***al preparato****.*
 'Add the eggs to the mixture'.

(3) a. ***Prendi*** *100 g di acqua ghiacciata.*
 'Take 100 g of cold water'.
 b. ***Versa*** *100 g di acqua ghiacciata.*
 'Pour 100 g of cold water'.

(4) a. *Metti i fiori di zucca nella* ***farina****.*
 'Put squash blossoms in the flour'.
 b. *Metti i fiori di zucca nella* ***pastella****.*
 'Put squash blossoms in the batter'.

In the first two examples, the clarification need is requested by the lack of specification. Specifically, in (1a) the verb is implicit, but without the direct formulation of the verb the system cannot be sure of what to do with the uttered ingredient; as a matter of fact, in combination with *eggs* other actions can be used, among the most frequent of which we can, for instance, mention *to beat*. The statistic analysis of the data can, furthermore, be used to infer user behaviours concerned with the fact that the verb is mostly left out when we have less specific actions in mind, such as *to take*, *to crack* or *to add*. For the reconstruction of these ellipsis human speakers conversely rely on the extra-linguistic context. In (2a) the argument *mixture* is not uttered, but the clarification is necessary for the system as in the previous example, as far as disambiguation purposes are concerned. On the other hand, the other two examples show a paradigmatic problem. In (3a) the verb *to take* is used instead of the verb *to pour*, being an implicit part of the action of pouring something. Nevertheless, its meaning is incomplete for the activation of the specific action and needs, therefore, a clarification. Finally, in (4a-b) a meronymic relation-linked vocabulary choice caused the uttering of a raw ingredient to designate the mixture it is part of. The inference and the consequent learning of these kinds of relationships between incomplete or imprecise commands and more precise ones is important to ensure an interactive learning module starting from the data.

3.2 Ontology-Based Learning

As far as what we call ontology-based learning is concerned, the template extracted from the data, and to which annotations are related, can be used as a ready to be extended data structure with the predefined use of a database. For this purpose, we mention a linguistic resource, such as MultiWordNet-Extended [9], to automatically extend the vocabulary collected in the corpus, thus enabling the system to understand examples which are not directly present in it. As a matter of fact, the interaction here is intended as the requests communication between the data and the aforementioned database.

(5) a. *Pelare 2 kg di patate. = sbucciare(due kg, patate)*

'Peel 2 kg of potatoes'. = peel(two kg, potatoes)

b. *Sbuccia 2 mele. = sbucciare(due, mele)*

'Peel two apples'. = peel(two, apples)

In (5a), for instance, the argument-slot occupied by *potatoes* can be paradigmatically substituted with another term, such as *apples*, belonging to the same semantic field *Gastronomy*. This is possible by linking the argument position (*ingrediente*),[6]

[6]The hypernym *ingrediente* (En. *ingredients*) is chosen to retrieve all the appropriate hyponyms and synonyms which are semantically related to the argument uttered in the example.

Fig. 2 Graph representing the ingredients used in the preparation of pancakes without butter (It. *pancakes senza burro*), extracted from *Giallo Zafferano*. The pancake recipe is represented as the central node; the ingredients extracted are 'farina' (En.*flour*), 'olio di oliva' (En. *olive oil*), 'uova' (En. *eggs*), 'lievito in polvere per dolci' (En. *baking powder*), 'yogurt', 'latte' (En. *milk*); the relationship connecting the central node (recipe) and the other nodes (ingredients) is USES

from the correspondent intent template, to the database via a specific query,[7] as in (1), to extract the required lexical unit.

```
MATCH (n:NOUN {word:'ingrediente'})-[:BELONGS_TO]->(m
    :SYNSET)-[:BELONGS_TO]->(j:SEMANTIC_FIELD {name: '
    Gastronomy'})<-[:BELONGS_TO]-(b:SYNSET)<-[:
    BELONGS_TO]-(w:NOUN) RETURN n.word, w.word
```

This query generates a list of 944 nouns, among which we mentios as instances 'candito' (En. *candid fruit*), 'caramello' (En. *caramel*), 'carciofo' (En. *artichoke*), etc. This automatic extraction of data simplifies the collection phase. A work in progress is directed to the implementation of a specific database of recipes which connects recipes' name to their specific ingredients (Fig. 2), quantities, and tools with specific relationship, in order to reduce possible erroneous extractions from a generic semantic database.

[7] As MultiWordNet-Extended is a graph database implemented in Neo4J [17], we are here referring to Cypher queries.

4 Conclusion

In this paper, we presented the SUGAR Corpus, its collection and its annotation. Furthermore, we focused our attention to the interactive structure of the dataset and the learnability techniques which can be applied by using it, especially as far as both clarification-based and ontology-based learning are concerned. The corpus as it is aims therefore at reducing the cost of the online learning processes. Next developments include the enlargement of the corpus and of the database and the creation of a dataset for the English language, alongside with the introduction of multimodality. As a matter of fact, pointing gestures or mimed actions and movements are multimodal activities interesting for this field of application as for any other spoken understanding task where a shared context of interaction is expected.

Acknowledgements This work is funded by the Italian PRIN project *Cultural Heritage Resources Orienting Multimodal Experience* (CHROME) #B52F15000450001.

References

1. Caselli T, Novielli N, Patti V, Rosso P (2018) Evalita 2018: overview of the 6th evaluation campaign of natural language processing and speech tools for italian. In: Caselli T, Novielli N, Patti V, Rosso P (eds) Proceedings of sixth evaluation campaign of natural language processing and speech tools for Italian. Final workshop (EVALITA 2018). CEUR.org, Turin, Italy
2. Di Maro M, Origlia A, Cutugno F (2018) Overview of the evalita 2018 spoken utterances guiding chef's assistant robots (sugar) task. In: Proceedings of EVALITA 2018
3. Forsythand EN, Martell CH (2007) Lexical and discourse analysis of online chat dialog. In: International conference on semantic computing, 2007. ICSC 2007. IEEE, pp 19–26
4. Godfrey JJ, Holliman EC, McDaniel J (1992) Switchboard: telephone speech corpus for research and development. In: [Proceedings] ICASSP-92: 1992 IEEE international conference on acoustics, speech, and signal processing, vol 1, pp 517–520. doi:https://doi.org/10.1109/ICASSP.1992.225858
5. Henderson J, Lemon O, Georgila K (2008) Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. Comput Linguist 34(4):487–511
6. Lemon O, Georgila K, Henderson J, Stuttle M (2006) An isu dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the talk in-car system. In: Proceedings of the eleventh conference of the european chapter of the association for computational linguistics: posters and demonstrations, pp 119–122. Association for Computational Linguistics
7. Li J, Miller AH, Chopra S, Ranzato M, Weston J (2016) Learning through dialogue interactions by asking questions. arXiv preprint arXiv:1612.04936
8. Liu B, Tur G, Hakkani-Tur D, Shah P, Heck L (2018) Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. arXiv preprint arXiv:1804.06512
9. Origlia A, Paci G, Cutugno F (2017) Mwn-e: a graph database to merge morpho-syntactic and phonological data for Italian. In: Proceedings of subsidia, page to appear
10. Paek T (2006) Reinforcement learning for spoken dialogue systems: comparing strengths and weaknesses for practical deployment. In: Proceedings of the dialog-on-dialog workshop, interspeech

11. Raux A, Langner B, Bohus D, Black AW, Eskenazi M (2005) Let's go public! taking a spoken dialog system to the real world. In: Ninth European conference on speech communication and technology
12. Rieser V, Kruijff-Korbayová I, Lemon O (2005) A corpus collection and annotation framework for learning multimodal clarification strategies. In: 6th SIGdial workshop on discourse and dialogue
13. Rieser V, Lemon O (2006) Cluster-based user simulations for learning dialogue strategies. In: Ninth international conference on spoken language processing
14. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2018) A survey of available corpora for building data-driven dialogue systems. J Vers Dial Discour 9(1):1–49
15. Tiedemann J (2012) Parallel data, tools and interfaces in opus. Lrec 2012:2214–2218
16. Tur G, Deng L (2011) Intent determination and spoken utterance classification. Spoken language understanding: systems for extracting semantic information from speech, pp 93–118
17. Webber J, Robinson I (2018) A programmatic introduction to neo4j. Addison-Wesley Professional
18. Young S, Gašić M, Thomson B, Williams JD (2013) Pomdp-based statistical spoken dialog systems: a review. Proc IEEE 101(5):1160–1179

Framing Lifelong Learning as Autonomous Deployment: Tune Once Live Forever

Eneko Agirre, Anders Jonsson, and Anthony Larcher

Abstract Lifelong Learning in the context of Artificial Intelligence is a new paradigm that is still in its infancy. It refers to agents that are able to learn continuously, accumulating the knowledge learned in previous tasks and using it to help future learning. In this position paper we depart from the focus on learning new tasks and instead take a stance from the perspective of the life-cycle of intelligent software. We propose to focus lifelong learning research on autonomous intelligent systems that sustain their performance after deployment in production across time without the need of machine learning experts. This perspective is being applied to three European projects funded under the CHIST-ERA framework on several domains of application.

1 Introduction

In the context of Artificial Intelligence, intelligent systems based on machine learning rely on human experts across all phases of the software. According to the de-facto standard CRISP-DM methodology (Cross-industry standard process for data mining [1]), the life-cycle of such software includes a development cycle with interactions to define the business domain, understand and prepare the data, select and train

E. Agirre (✉)
Ixa NLP Group, University of the Basque Country UPV/EHU, Manuel Lardizabal 1, 20018 Donostia, Spain
e-mail: e.agirre@ehu.eus

A. Jonsson
Department of Information and Communication Technology, Universitat Pompeu Fabra, Roc Boronat 138, 08018 Barcelona, Spain
e-mail: anders.jonsson@upf.edu

A. Larcher
LIUM - Le Mans Université, Le Mans, France
e-mail: anthony.larcher@univ-lemans.fr

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_29

appropriate models, and evaluate the software. After several iterations, development is halted and the system is deployed in production.

Such systems are constructed, developed and trained until a certain point of time, and once deployed for production in the target environment they need to be continuously monitored to check that its performance in real life matches the one in development. Current intelligent systems are not able to adapt to situations outside their development conditions. For this reason, the flow of real data and system responses needs to be monitored, to ensure that it follows the same distribution as the data used in development, as otherwise the performance of the intelligent system will degrade with time. Usually human monitoring from field experts is needed to detect whether performance degrades, and at that point, development experts (e.g. machine learning experts) are needed to revise the system and adapt it to the new conditions. In short, intelligent systems are not able to learn from the environment and improve themselves without intervention from their initial developers.

Lifelong Learning in the context of Artificial Intelligence[1] is a new paradigm that is still in its infancy. It is commonly defined as agents that are able to learn continuously, accumulating the knowledge learned in previous tasks and using it to help future learning [3]. The concept of "new task" is not well defined, as it has been used for different phenomena like new instances unseen in training, related tasks or new domains, but it is generally demonstrated on toy tasks [2, 4]. We depart from the focus on learning new tasks and, instead, take a stance from the perspective of the life-cycle of intelligent software that addresses real tasks. We propose to focus lifelong learning research on autonomous intelligent systems that sustain their performance across time *after deployment* without the need of machine learning experts, following the "*Tune once, live forever*" motto. We think that such a capability of incremental autonomous learning, or lifelong learning, is key to the development of truly autonomous intelligent systems, as argued in the CHIST-ERA 2016 call for projects.[2]

In this position paper we first detail our proposal, and then mention how we plan to address this issue in three European projects funded under the CHISTERA framework.

2 Lifelong Learning for Autonomous Deployment

Given a task, the final goal is to design a system which is able to continue learning after deployment, adapting itself to the changes in the distribution of the input data when necessary, without the need of further development. Figure 1 shows two possible scenarios. In the first scenario (left of figure) the lifelong learning system (LL system for short) is able to learn from a stream of data over time, improving the performance

[1] Also called continual learning [2], among other terminological options.

[2] http://www.chistera.eu/sites/chistera.eu/files/CHIST-ERA%20Call%202016%20-%20Call%20Announcement.pdf.

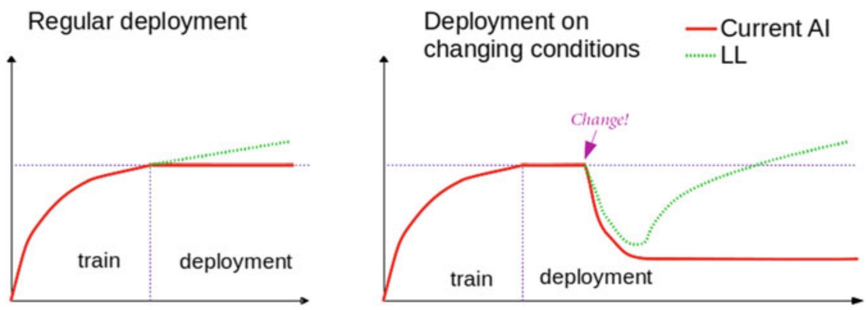


Fig. 1 Lifelong learning under two scenarios. In the left, the distribution of input data does not change with time after deployment. In the right, the distribution of input data changes after deployment

over time. The right side of the figure shows a more realistic scenario, where, at some point of time, there are changes in the distribution of the input data, and the system performance drops. The LL system (in green) will be able to recognize the change in the environment and take actions to adapt and improve performance, reaching, or even surpassing, the original performance level. The figure also shows a current AI system (in red), which is not able to adapt.

We envision that such LL systems need to solve two key challenges:

Auto-evolution: An LL system needs to identify relevant new information that was not present at the time of development, and integrate this information into its learning model. An LL system also needs to balance the importance of old and new data, since an excessive focus on old data may prohibit the system from adapting to changes, while an excessive focus on new data may lead to catastrophic forgetting.

Auto-evaluation: An LL system needs to adapt to changes in the underlying input distribution, which may require modifying the objective function. For example, in supervised learning a system needs to automatically select new inputs and associate labels with these inputs, which implicitly modifies the objective function to include a measure of how well the system covers these new examples.

Although the final goal is a fully autonomous system, a weaker system which only requires field experts would also represent an important breakthrough, as it would lower the reaction time and costs of the current model, where the interaction with developers is also needed. In this weaker case, field experts might help recognizing the degradation of performance, gathering relevant adaptation data and label it when necessary.

Another key challenge is evaluation. Metrics and protocols for traditional machine learning task bench-marking have been thoroughly developed for decades through intensive efforts and numerous evaluation campaigns. Adapting them to match the LL paradigm is a necessary step to catalyse the development of autonomous systems and provide means of reproducible and fair evaluation.

We have applied the perspective outlined in this section to three European Projects: ALLIES,[3] DELTA[4] and LIHLITH.[5]

3 Approach in Three Heterogeneous Projects

In this Section we present three different approaches for LL systems as envisioned in three projects: ALLIES, DELTA and LIHLITH.

The ALLIES project envisions LL systems guided by domain experts through active learning and continuous supervision. The human in the loop is expected to provide a safeguard across time but also make sure the adaptation policy is matching the customer needs. A key aspect of ALLIES lays in the supervised evaluation of such systems which performance must be considered according to the human effort provided across time. Implementations of machine translation and speaker diarization autonomous systems will help design the requested metrics and protocols. The main outcomes of the project consist of a secured platform to evaluate autonomous systems in a reproducible manner and two bench-marking campaigns that will validate the evaluation scenarios.

The DELTA project aims at developing novel algorithms for lifelong reinforcement learning, an area of machine learning that studies sequential decision making. Concretely, the objective of DELTA is to address three key problems related to lifelong reinforcement learning, and develop solutions that improve on the state-of-the-art: (1) plan high-quality sequences of actions; (2) efficiently explore the environment; and (3) decompose the overall task into subtasks. Auto-evolution will be tested by changing observable variables and available actions, and measuring how robust a system is to such changes. Auto-evaluation and benchmarking are already part of reinforcement learning, in the sense that the evaluation metric takes the form of a reward signal that the system has to learn to maximize. Since reinforcement learning is online by nature, a system constantly improves over time, and the balance of old and new data is controlled by tuning a learning rate. However, the reward is usually defined by a human expert, while in the lifelong setting the system itself has to define the reward associated with new, previously unseen tasks.

The LIHLITH project applies lifelong learning to the interaction of humans and machines on specific domains. LIHLITH focuses on human-computer dialogue, where each dialogue experience is used by the system to learn to better interact, based on the success (or failure) of previous interactions. The dialogues are designed to produce a reward, allowing the chatbot system to know whether the interaction was successful or not. The reward is used to train the domain and dialogue management modules of the chatbot, improving the performance, and reducing the development cost, both on a single target domain but specially when moving to new domains.

[3] https://projets-lium.univ-lemans.fr/allies/.

[4] https://www.upf.edu/web/delta.

[5] http://ixa2.si.ehu.eus/lihlith/.

Contrary to other domains, in human-machine dialogue auto-evaluation is more natural, in the sense that user feedback can be naturally blended in the conversation, providing new learning instances. Auto-evolution can be triggered when detecting that negative feedback is associated with patterns in the data (e.g. change of domain). Please see [5] for a recent successful example, where simple patterns are used to get feedback in the framework of simple chatbots.

4 Relation to Recent Work on Lifelong Learning

Research in lifelong reinforcement learning has mainly focused on learning strategies for solving a set of tasks [6, 7]. However, in this setting the set of tasks is usually fixed and known. There also exists work on transfer learning that investigates how an existing strategy can be adapted to solve a new task [8]. Preliminary work in DELTA has studied how exploration can be adapted to the case for which model components change over time [9]. There are also preliminary results investigating the theoretical properties of hierarchical decomposition [10, 11], but this work assumes that the decomposition is already given rather than inferred by the system.

5 Conclusion and Discussion

In this position paper we depart from the focus on learning new tasks and instead take a stance from the perspective of the life-cycle of intelligent software. We propose to focus lifelong learning research on autonomous intelligent systems that sustain their performance after deployment in production across time without the need of machine learning experts. We briefly report how this perspective is applied to three European projects funded under the CHIST-ERA framework on several domains of application.

Acknowledgements This work has been partially funded by the ALLIES, DELTA and LIHLITH projects supported by the EU ERA-Net CHIST-ERA and the Spanish Research Agency (LIHLITH, PCIN-2017-118; DELTA, PCIN-2017-082).

References

1. Shearer C (2000) The CRISP-DM model: the new blueprint for data mining. Data Warehous 5:13–22
2. Sodhani S, Chandar S, Bengio Y (2018) On training recurrent neural networks for lifelong learning. CoRR, arXiv:1811.07017

3. Chen Z, Liu B, Brachman R, Stone P, Rossi F (2018) Lifelong machine learning: second edition. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool Publishers
4. Li J, Miller AH, Chopra S, Ranzato MA, Weston J (2017) Learning through dialogue interactions by asking questions. In: ICLR
5. Hancock B, Bordes A, Mazaré P-E, Weston J (2019) Learning from dialogue after deployment: feed yourself, chatbot! CoRR, arXiv:1901.05415
6. Bou Ammar H, Tutunov R, Eaton E (2015) Safe policy search for lifelong reinforcement learning with sublinear regret. In: Proceedings of the international conference on machine learning (ICML'15), 2015
7. Tessler C, Givony S, Zahavy T, Mankowitz D, Mannor S (2017) A deep hierarchical approach to lifelong learning in minecraft. In: Proceedings of the AAAI conference on artificial intelligence (AAAI'17)
8. Abel D, Jinnai Y, Guo Y, Konidaris G, Littman M (2018) Policy and value transfer in lifelong reinforcement learning. In: Proceedings of the international conference on machine learning (ICML'18)
9. Abel D, Jinnai Y, Guo Y, Konidaris G, Littman M (2018) Policy and value transfer in lifelong reinforcement learning. In: Proceedings of the international conference on machine learning (ICML'18)
10. Brunskill E, Li L (2014) Pac-inspired option discovery in lifelong reinforcement learning. In: Proceedings of the international conference on machine learning (ICML'14)
11. Brunskill E, Li L (2014) Pac-inspired option discovery in lifelong reinforcement learning. In: Proceedings of the international conference on machine learning (ICML'14)

Continuous Learning for Question Answering

Anselmo Peñas, Mathilde Veron, Camille Pradel, Arantxa Otegi, Guillermo Echegoyen, and Alvaro Rodrigo

Abstract We consider the problem of answering natural language questions over a Knowledge Graph, in the case of systems that must evolve over time in a production environment. One of the key issues is that we can expect that the system will receive questions that cannot be answered with the current state of the Knowledge Graph. We discuss here the challenges we need to address in this scenario and the expected behavior of this kind of Lifelong learning system. We also suggest a first task to address this problem and a possible procedure to build a benchmark.

1 Scenario

We focus on the problem of Question Answering (QA) over Knowledge Bases: the task of building an interpretation to a Natural Language (NL) question, according to

A. Peñas (✉) · G. Echegoyen · A. Rodrigo
Universidad Nacional de Educación a Distancia (UNED), Juan del Rosal 16, 28040 Madrid, Spain
e-mail: anselmo@lsi.uned.es

G. Echegoyen
e-mail: gblanco@lsi.uned.es

A. Rodrigo
e-mail: alvarory@lsi.uned.es

M. Veron
LIMSI, CNRS, Université Paris-Saclay, Campus universitaire, bât. 507 rue du Belvédère, 91405 Orsay Cedex, Paris, France
e-mail: mathilde.veron@u-psud.fr

C. Pradel
Synapse Développement, 7 boulevard de la Gare, 31500 Toulouse, France
e-mail: camille.pradel@synapse-fr.com

A. Otegi
Ixa NLP Group, University of the Basque Country UPV/EHU, Manuel Lardizabal 1, 20018 San Sebastian, Spain
e-mail: arantza.otegi@ehu.eus

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_30

a Knowledge Base (KB), Database (DB) or Knowledge Graph (KG). Without much loss of generality, we will assume the KB has the form of a KG for the purposes of our study.

The general task is defined as follows: Given a NL question, produce a query in a formal language (e.g. SPARQL) to return from the KG the correct answers to the question.

In this setting, the aim of this work is to study the problem of answering natural language questions over a Knowledge Graph, in the case of systems that must evolve over time in a production environment. One of the key issues is that we can expect that the system will receive questions that cannot be answered with the current state of the Knowledge Graph.

The first challenge, then, is the detection of this situation: detect if the user's utterance can be mapped or not into an explicit knowledge subgraph. Maybe the answer is not in the KG because of a lack in its population (i.e. some resources/instances or property values are missing). But in the general case, it could happen that the KG doesn't contain information about classes and properties that might be of new interest for the user. In this case, a good system, which can evolve over time and adapt to these "surprise" new situations, should be able to look for resources outside, in order to extract and populate the KG, not only with new instances but also new classes and new properties.

This problem has already captured the attention of some researchers such as Mazumder and his colleagues [1] who presented the LiLi system, although in that work, the problem is only addressed partially and without a previous definition of the complete general scenario. In particular, queries to the LiLi system are just single triples, reducing to the trivial case the problem of deciding whether the answer to a question is in the KG or not. It simplifies also the problem of detecting the pieces of knowledge that have to be added to the KG. The option taken for LiLi system for enriching the KG is to ask the user for some missing pieces of knowledge and try to find strategies to infer some others.

However, in the general scenario of complex NL Question-Answering over KGs these decisions are not trivial. If a system does not get an answer to a question, it could be due to several factors, including some errors in the process of NL interpretation (e.g. Entity Linking).

We are agnostic with respect to the strategy and the role of users to address this problem. But this is a hard problem and we envisage that it should be solved in collaboration with users, especially for the explicit or implicit assessment of new information correctness, and the decision to incorporate it into the KG.

For the purposes of clarifying the scenario, consider the following example: Assume a cooking scenario where we have classes, such as recipe or ingredient, with instances such as mousse or milk, respectively, together with properties such as a has ingredient relation between objects of recipe class and objects of ingredient class. Imagine now that the user asks something that cannot be answered with the current state of the KG: "How long do I need to put my mousse in the fridge?". We expect a system to follow a procedure similar to:

1. The system must determine that it cannot answer this question with the KG.
2. Then, the system needs to answer this question by making use of external resources (it can be done by collaborating with a user [4, 5], after processing a large document collection [6], by querying the web [7], or through a combination of them). Imagine that finally the piece of text accepted says: "Chill the mousse in the refrigerator for 2 h".
3. The system must identify that the KG has not an explicit relation between a recipe and a time duration similar to "time in the fridge". So, it decides to declare a new relation "fridge-duration" with range on recipes and domain on time duration.
4. The system adds a new triple for this relation: <mousse, fridge-duration, "P0DT2H"^^xsd:duration>
5. Once the system has confirmation of a relation that must be added to the KG, together with a good example, then, the system must go through all instances of the involved class (i.e. recipes) and use the external sources to try to populate this new relation in the KG. Depending on the approach, it could be interesting for the system to maintain other information such as the natural language statements used to express the relation, etc.

This is a very ambitious scenario that poses many interesting research questions that can be addressed in many different ways, with many different strategies. The first effect is that current state of the art in QA over KGs cannot address this problem because most of the systems are built over the assumption that the KG contains the answer to the question [2, 3].

Therefore, in our opinion, this is the first challenge we should address in the described scenario: the detection of situations where the KG does not contain the required pieces of information.

2 A Proposal for a Shared Task

According to the scenario described above, we propose to address an initial task as the first step towards the development of continuous-learning QA systems. Since we are working with KGs, we can assume without loss of generalization, that we have objects, types or classes attached to the objects, literals, and relations or properties between objects and literals. Therefore, the missing pieces of information that impede the mapping of a user utterance into a KG subgraph can be any of them.

The task, then, is: Given a KG, and an external resource such as a text collection in the same domain:

1. decide whether a user utterance in Natural Language can be mapped or not into the KG and, if it is not possible,
2. determine through which ones of the following ways the system must find a strategy to enrich the knowledge graph:

a. Declare a new class (or object type). For example: utensil.
b. Add a new instance of a class. For example: rolling pin.
c. Declare a new relation. For example: needs_utensil.
d. Add a new triple to the KG. For example: <shortbread, needs_utensil, rolling pin>.

The task is being expressed as a classification problem, which allows a wide range of different techniques. At the end, we would be training a set of five binary classifiers (or a multi-label classifier).

To test this task, we would need a benchmark dataset. There are several ways in which we can build such dataset, but perhaps the easiest one is to take advantage of current QA datasets with NL questions that must be answered according to a KG. For each question we can remove some elements form the KG that impede the system to answer the question. These elements can be from a particular instance up to a complete relation. Thus, somehow what we are expecting from the system is to guess the kind of information that is missing in the KG.

This proposal is related to some approaches developed by other researchers for evaluating their proposals in the context of Lifelong Learning. For example, [8] evaluated the ability of their system adding new triples to a KG, but not new classes or properties. On the other hand, [1] evaluate the ability of their system finding strategies for including new knowledge including relations. However, their system works directly over triples and not over NL questions. Thus, the decision about whether the current state of the KG permit to answer or not the question is trivial. This decision has a strong inter-relation with the problem of NL understanding and should be considered in this context.

Acknowledgements This work has been funded by the Spanish Research Agency (PCIN-2017-085, PCIN-2017-118/AEI, Spain) and the Agence Nationale pour la Recherche (ANR-17-CHR2-0001-03, ANR-17-CHR2-0001-04/ANR, France) with the LIHLITH project inside the ERA-Net CHIST-ERA framework.

References

1. Mazumder S, Ma N, Liu B (2018) Towards a continuous knowledge learning engine for chatbots. arXiv:1802.06024
2. Hoffner K, Walter S, Marx E, Usbeck R, Lehmann J, Ngonga Ngomo A-C (2017) Survey on challenges of question answering in the semantic web. Semant Web 8(6):895–920

3. Diefenbach D, Lopez V, Singh K, Maret P (2018) Core techniques of question answering systems over knowledge bases: a survey. Knowl Inf Syst 55(3):529–569
4. Li J, Miller AH, Chopra S, Ranzato M, Weston J (2016) Learning through dialogue interactions by asking questions. arXiv:1612.04936
5. Weston JE (2016) Dialog-based language learning. Adv Neural Inform Process Syst 829–837
6. Mazumder S, Liu B (2017) Context-aware path ranking for knowledge base completion. IJCAI
7. West R, Gabrilovich E, Murphy K, Sun S, Gupta R, Lin D (2014) Knowledge base completion via search-based question answering. In: Proceedings of the 23rd international conference on World wide web. ACM, pp 515–526
8. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka Jr ER, Mitchell, TM (2010) Toward an architecture for never-ending language learning. In: Proceedings of the 24th conference on artificial intelligence. AAAI Press

Live and Learn, Ask and Tell: Agents over Tasks

Don Perlis, Clifford Bakalian, Justin Brody, Timothy Clausner, Matthew D. Goldberg, Adam Hamlin, Vincent Hsiao, Darsana Josyula, Chris Maxey, Seth Rabin, David Sekora, Jared Shamwell, and Jesse Silverberg

Abstract Steven Spielberg's "A.I." tells the story of two artificial agents: David and Teddy. While David resembles a human child, his companion Teddy is much simpler. Its behavior, however, still suggests a crucial mix of capabilities that stretch the state of the art in AI today. We argue that, unlike most contemporary AI, Teddy qualifies as a bona fide *agent*, and that implementing such a system would represent a valuable advance in our understanding of agency. We then describe a project to integrate our existing work to create a simple agent with Teddy-like capabilities.

Teddy learns by dialog as well as observation, and makes helpful comments based on its assessment of other agents. A real-life Teddy would seem to involve a synergistic integration of multiple areas including robotics, vision, learning, NLP, knowledge representation, and commonsense and multi-agent reasoning. Teddy doesn't need to *do* a whole lot in the ordinary sense. It is there learning/knowing more and more all the time, is available for simple interactions (such as asking and telling), and can offer suggestions and remember events as they transpire over time (e.g., who else was present on a particular occasion).

D. Perlis · C. Bakalian · T. Clausner · M. D. Goldberg · A. Hamlin · V. Hsiao · C. Maxey · S. Rabin · D. Sekora (✉) · J. Shamwell · J. Silverberg
University of Maryland, College Park, MD 20742, USA
e-mail: djsekora@cs.umd.edu

D. Perlis
e-mail: perlis@umd.edu

J. Brody
Franklin and Marshall College, Lancaster, PA 17603, USA

D. Josyula
Bowie State University, Bowie, MD 20715, USA

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_31

By contrast, a great deal of AI is task-based: write a program that accepts input type I and produces output type O. The term "agent" is sometimes used loosely for almost any AI program. But, as we intend the term here, an agent (e.g., Teddy) is an entity that knows what tasks it is attempting, and when and why, and can reason and make decisions about them while doing them, as well as before and after. What is of interest here is not that Teddy succeeds at a particular task X, but that Teddy knows that it its attempting X, and can reason about it and remember it later.

Here we summarize several relevant advances which we are now engaged in integrating into such a (robotic-dialogic) agent:

Anomaly-handling: We designed and implemented the Metacognitive Loop (MCL) with the purpose of allowing a system to handle anomalies [2, 4] by (i) *noting* when the system's actions do not line up with its expectations, (ii) *assessing* any such anomaly and examining the available routes for resolving it, and (iii) *guiding* the system toward implementing the chosen strategy. This was built using our Active Logic Machine [3], which is a paraconsistent and temporal reasoning engine. [12] discusses how agent-centric logics, such as active logic, are well suited for resolving contradictions through introspection due to the temporal stratification of the inference process. MCL in a way embodies an implementation of the appearance-reality distinction [8]: things may be different from what the agent has believed them to be. This leads to a need for representation of one's past beliefs. Active logic is used for this purpose, for instance to create expressions for "the thing I formerly took to be X" in the case of misidentification [11]. **Dialog and intention:** [13] describes ways that active logic can be used to prevent implications that could misleadingly impact the intended meaning of input phrases. In [10], the TRAINS system [1] is extended using MCL to determine if it understood a speech request and then takes actions to correct errors. **Dialog and robotic self-knowledge:** We sketched a robotic activity: on request, a robot goes to another room to find a particular book [5]. This may sound simple enough, but it is surprisingly complex when designed in a general way. Two robots (Alice and Julia) explored this and unexpectedly brought us face-to-face with new issues. In particular, when Alice was programmed to point at Julia while saying that it was doing so, it would enter an unintended loop by responding to its own utterance of "Julia". Using the neuroscientific idea of efference copy, we enabled Alice to know when she was engaged in talking and thus not respond [7]; this differs markedly from the approach of [6] who programmed a robot to recognize the sound of its own voice, and then decide on that basis that it must be speaking. **Dialog and self/other knowledge:** We recently examined self/other knowledge complexities arising even in a simple dialogue [9]. If asked, "Is there milk in the fridge?" Teddy should: (i) realize another agent is addressing it, (ii) consider if it has appropriate knowledge of the fridge and milk, or (iii) if it realizes it doesn't know, then possibly (iv) infer that it can find out by looking in the refrigerator, and so on. Finally, after responding, it should understand that the other agent now knows the information in the response. **Navigation:** We have developed a voice-controlled navigation interface for our robot agent. Current work includes: (1) expanding the command vocabulary to include more complex movements and objectives; (2) using computational linguistics to attempt to resolve unknown/misheard commands. **Object recognition:** We are using object-detection software from state of the art deep neural networks to locate and classify objects within an image. Our network can detect bottles, chairs, people, and other common indoor objects. This can be made more complex via different neural network architectures, increased network object classes, and eventually specific agent identification (or misidentification leading to anomaly detection and response). *We expect that a Teddy-bot endowed with a variety of abilities, particularly learning, asking, and telling, operating over its lifetime, will be able to assess and adjust its actions, and, in so doing, will yield new discoveries about how dialog and lifelong learning enable task-general behaviors.*

References

1. Allen JF, Miller BW, Ringger EK, Sikorski T (1996) A robust system for natural spoken dialogue. In: Proceedings of the 34th annual meeting on association for computational linguistics, pp 62–70. Association for Computational Linguistics
2. Anderson ML, Perlis DR (2005) Logic, self-awareness and self-improvement: the metacognitive loop and the problem of brittleness. J Logic Comput 15(1):21–40
3. Anderson ML, Gomaa W, Grant J, Perlis D (2008) Active logic semantics for a single agent in a static world. Artif Intell 172(8–9):1045–1063
4. Anderson ML, Oates T, Chong W, Perlis DR (2006) The metacognitive loop I: enhancing reinforcement learning with metacognitive monitoring and control for improved perturbation tolerance. J Exper Theor Artif Intell 18(3):387–411
5. Bhargava P, Cox M, Oates T, Oh U, Paisner M, Perlis D, Shamwell J (2012) The robot baby and massive metacognition: future vision. In: Proceedings of the 2012 IEEE international conference on development and learning and epigenetic robotics, pp 1–2
6. Bringsjord S, Licato J, Govindarajulu NS, Ghosh R, Sen A (2015) Real robots that pass human tests of self-consciousness. In: 2015 24th IEEE international symposium on robot and human interactive communication (RO-MAN), pp 498–504. IEEE
7. Brody J, Perlis D, Shamwell J (2015) Who's talking–efference copy and a Robot's sense of agency. In: 2015 AAAI Fall symposium series
8. Flavell JH, Green FL, Flavell ER, Watson MW, Campione JC (1986) Development of knowledge about the appearance-reality distinction. Monograph Soc Res Child Develop 51(1):i–87
9. Goldberg MD, Brody J, Clausner TC, Perlis D (2018) The overlooked role of self-agency in artificial systems. In: Workshop on shortcomings in vision and language at european conference on computer vision
10. Josyula DP, Fults S, Anderson ML, Wilson S, Perlis D (2007) Application of MCL in a dialog agent. In: Third language and technology conference
11. Miller M (1993) A view of one's past and other aspects of reasoned change in belief. University of Maryland at College Park
12. Perlis D, Brody J, Kraus S, Miller M (2017) The internal reasoning of robots. In: Proceedings of the thirteenth international symposium on commonsense reasoning
13. Purang K, Perlis D, Gurney J (1996) Active logic applied to cancellation of gricean implicature. In: AAAI 96 Spring symposium on computational implicature

Lifelong Learning and Task-Oriented Dialogue System: What Does It Mean?

Mathilde Veron, Sahar Ghannay, Anne-Laure Ligozat, and Sophie Rosset

Abstract The main objective of this paper is to propose a functional definition of lifelong learning systems adapted to the framework of task-oriented dialogue systems. We mainly identified two aspects where a lifelong learning technology could be applied in such systems: to improve the natural language understanding module and to enrich the database used by the system. Given our definition, we present an example of how it could be implemented in an existing task-oriented dialogue system that is developed in the LIHLITH project.

1 Introduction

A dialogue system allows a user to interact using natural language. Two families of dialogue systems exist: conversational systems and task-oriented systems. Conversational systems have to generate the most appropriate reaction given a user's utterance and a context, without any restriction about the domain. A task-oriented system aims to help the user perform a task or access information (in this family, both open and limited domains are possible). A dialogue system generally consists

M. Veron · S. Rosset (✉)
LIMSI, CNRS, Université Paris-Saclay, Campus universitaire, bât. 507,
91405 Orsay Cedex, France
e-mail: sophie.rosset@limsi.fr

M. Veron
e-mail: mathilde.veron@limsi.fr

S. Ghannay
LIMSI, CNRS, Université Paris-Sud, Université Paris-Saclay, Campus universitaire,
bât. 507, 91405 Orsay Cedex, France
e-mail: sahar.ghannay@limsi.fr

A.-L. Ligozat
LIMSI, ENSIIE, CNRS, Université Paris-Saclay, Campus universitaire, bât. 507,
91405 Orsay Cedex, France
e-mail: anne-laure.ligozat@limsi.fr

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_32

of three modules: natural language understanding (NLU), dialogue management and natural language generation (NLG). In this paper, we focus on task-oriented system, and more precisely, on classical task-oriented systems and not on end-to-end ones.

There has recently been a growing interest for building lifelong learning (LL) dialogue systems. LL is a subject which has led to numerous works in different fields, such as machine learning, deep learning,[1] artificial intelligence or robotics [2, 9, 11]. It is commonly considered as the "ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences" [9]. Performing LL in each field differs, depending on the nature of the new knowledge to learn, and on the way the learning is conducted. In the field of dialogue systems, the application of LL should result in dialogue systems which are able to continuously learn over time, by interacting with the user as two humans would do it in real life. For example, a human can learn new knowledge at any time by asking questions when facing an unknown situation, or simply by inferring new knowledge from the current conversation. A human is also able to use this learned knowledge directly or in future conversations. The new piece of knowledge can be for instance vocabulary, information or even an appropriate behavior to have in a specific situation. A LL dialogue system should thus be able, as a human, to improve its communication abilities and, in the case of a task-oriented system, to learn new knowledge specific to its application domain. This implies that the system should be able to detect new knowledge in the user's utterances and to understand when it should ask questions in order to learn new knowledge.

We therefore want to delimit how far a dialogue system can mimic the human continuous learning process, by determining exactly what a dialogue system can learn, how far it can *continuously* learn *over time*, and the methods which have to be developed to achieve this goal. In more simple terms, we want to determine what could be a LL dialogue system. It will also be interesting to wonder if we can adapt LL techniques from other fields to dialogue systems.

We present in Sect. 2 previous works which have been done about LL, reinforcement learning and learning dialogue systems. In Sect. 3 we attempt to define what a LL dialogue system is. Finally, we apply this definition to an existing cooking chatbot and try to propose mechanisms in order to perform LL, in Sect. 4.

2 Related Work

Lifelong learning (LL) has been of growing interest for several years resulting in numerous works and writings [2, 9, 11]. In their book [2], Chen and Liu define three features characterizing a LL system: continuous learning, knowledge accumulation and maintenance in the knowledge base (KB), and the ability to use past knowledge to help future learning. Moreover, they consider that a LL system should be able to

[1] See for example the workshop Continual Learning at NIPS in 2018, https://sites.google.com/view/continual2018/home?authuser=0.

detect new situations in the usage process and to adapt its behaviour to these new situations, by learning new tasks and performing self-motivated learning. The learned knowledge, which is accumulated, should also help the system to learn new tasks with little data or effort. In particular, the new task can belong to an other domain (subject area). However, this *new task* concept is unclear and seems to cover a broad variety of situations, going from *new instance* (i.e. a new instance of a concept in a model) to completely new domain [5].

Simultaneously, progress has been made in the domain of Reinforcement Learning (RL). A system using RL methods learns from past experiences to estimate the most accurate decision process by maximizing a predefined reward function. In particular, RL in dialogue systems has been widely studied to enable the system to learn a dialogue policy [13], considering for example online scenarios [13], and inverse RL framework [1]. More recently, Li and his colleagues [4] used the RL framework in their chatbot, which aims to answer questions of the movie domain. Thanks to RL, their system is able to learn when and how to ask questions to a user, in order to improve itself during this specific interaction.

Li and his colleagues [4] identify three situations where their dialogue system can ask questions:

- *Understanding problem:* the system asks for a paraphrase or verification. They focus here only on typos/spelling mistakes.
- *Knowledge operation problem:* the system needs help to perform reasoning steps over the KB.
- *Lack of knowledge:* the system asks for the answer.

RL methods can be applied to a dialogue system in order to perform LL but other methods are possible. For example, Mazumder and his colleagues developed a continuous knowledge learning engine for chatbots [6]. To achieve this, they focus on solving the open-world knowledge base completion problem by interactive learning and inference. They developed techniques to enable the system to continuously and interactively learn new knowledge in particular during real conversations. They use a knowledge store to retain the learned knowledge. They also developed a guessing mechanism, as they consider that the user may not know the information that the system is asking for.

In their second edition of their book [3], Chen and Liu dedicated the 8th chapter to continuous knowledge learning in chatbots. In this chapter, they essentially discuss the article mentioned above [6] and claim:

> Lifelong learning is reflected by the fact that the newly acquired facts are retained in the KB and used in inference for future queries, and that the accumulated knowledge in addition to the updated KB including past inference performances are leveraged to guide future interactions and learning.

However, Mazumder and his colleagues focus only on the problem of KB completion and provide a specific example of what can be a LL dialogue system. In Sect. 3, we propose a global definition of a LL dialogue system and we describe the different types of knowledge that can be learned.

3 What Is a LL Dialogue System: Attempt of a Definition

Before defining what a LL dialogue system is, we have to mention that LL should not be considered as a method like Reinforcement Learning, but more as a goal. In fact, the nature of the knowledge that can be learned, the methods involved and the objectives, strongly depend on the field and the application domain. It would though be interesting to define LL concepts and methods that can be shared between the different fields.

Thanks to the definitions given in [2, 3] and the previously described papers [4, 6], we can say that a lifelong learning dialogue system is a conversational system that continuously learns over time through interactions with a real user and possibly by asking questions. The learned knowledge should be retained, in order to allow the system to directly have access to this new knowledge, and to directly adapt its behaviour. The system should also keep the modifications for future dialogues. Moreover, the learned knowledge should help the system to infer additional knowledge and to learn in the future in a more efficient and in a faster manner. During interaction with the user, the system should thus be able to detect new situations and to learn how to adapt.

We now focus on task-oriented dialogue systems. This kind of system aims to help the user to perform a task or to access information. To achieve this, the system first tries to understand what the user is asking for, by calling the NLU sub-system which performs slot-filling and intent detection. Then, the dialogue manager tries to link the slots and the intent detected to the data that it has access to. Finally, the system returns the result by generating natural language.

Considering the previous definition of a LL dialogue system and the distinctions made in [4] we can see that a LL task-oriented dialogue system can learn at different levels:

- *Improve interactions*:
  - by being better at understanding what the user says (NLU)

    better perform slot-filling and intent detection
    be able to detect new intents and/or new slots
    be able to detect out-of-domain user's utterance

  - by being better at generating natural language (NLG)
- *Enrich data*: the system can learn new data by enriching its knowledge base
- *Link new intent to data*: the system can learn to link already known user's intent or new ones to the data it has access to. It can thus learn to perform new tasks.

However, we should pay particular attention to the fact that the user may transfer incorrect information, that can be learned by the system. For example, in the case of data enrichment, the system can ask the user about a specific piece of information that is missing in its knowledge base. The problem is that the user can not know the answer, can be wrong, or the data can be subjective. Moreover, he can also make typos or spelling mistakes, and he can have his own way to communicate given his age and where he lives for instance.

This consideration highlights, in particular, the need of defining when, or under which conditions, a learned knowledge should be considered as reliable. After this, the learned knowledge can be used for retraining a model, or can definitely be retained in the knowledge base of the system.

Moreover, to perform LL we could make use of *Online Reinforcement Learning* (RL) techniques, to allow the system to decide when it should ask questions and what questions it should ask, as done in [4]. All the methods which can be used rely on a certain amount of available data and even on data simulation (e.g. user simulation learned on available data in the case of RL).

4 Application to a Cooking Chatbot

As part of the LIHLITH project,[2] we developed a chatbot in the cooking domain, that allows, among others thing, a user to find a recipe matching his criteria.

In the following sections, we first give an overview of the system and then describe how LL could be implemented in it.

4.1 General Description of the LIHLITH Cooking Chatbot

The LIHLITH cooking chatbot is divided into three sub-systems:

- *NLU module:* takes as input the utterance of the user and returns the slots and the intent associated to this utterance. Considering the following user utterance: “Please find me a recipe of pancakes without eggs”, the NLU should detect the slots “recipe: pancakes” and “neg-ingredient: eggs” plus the intent “RECING”, that means that the user is looking for a recipe by giving the name of the recipe and the ingredients. The NLU module is based on deep neural network performing both slot-filling and intent detection [12], also known as *joint NLU*. We use an off-the-shelf system.[3] It has been trained with data which have been automatically generated and annotated thanks to patterns and lists of terms according to the method described in [7].
- *Dialogue manager:* decides what action to perform depending on the results of the NLU module and the current context, using a semantic textual similarity module or a database module.
- *NLG module:* generates the answer with the help of patterns

[2]Learning to Interact with Humans by Lifelong Interaction with Humans.

[3]All needed information can be found on https://github.com/SNUDerek/multiLSTM.

Note that both the dialogue manager and the NLG module are based on rule-based approaches.

The LIHLITH cooking chatbot[4] handles two different types of scenarios:

1. It searches in the recipes database if the user wants to find a recipe matching his criteria. The user may say for example “Please find me a recipe of pancakes without eggs”.
2. It searches in unstructured data by estimating semantic textual similarity, if the user asks a question relative to the cooking domain. The user may say for example “Why is -18% C the ideal freezer temperature?”.

For these two types of scenarios we use different types of data: the recipes database and the unstructured data. The recipes database has been built upon Wikipedia:Cookbook, and includes *1,064 recipes*. It contains for example information about the name, the details, the ingredients, the procedure and the categories. The unstructured data is composed of *784 non-recipe documents* that can be found on Wikipedia:Cookbook.

4.2 Lifelong Learning for Cooking Chatbot

Based on the definitions of Sect. 3, we decide to focus on two ways of performing LL on our chatbot: the improvement of understanding and the data enrichment. We discuss in the following subsections how it could be done, with the help of dialogue examples.

4.2.1 Improve Understanding

The first way aims at letting the system improve its understanding capabilities. In this case, it should be able to better detect slots and intents. It also should be able to learn new slots and new intents and to link each new intent to the right data.

There are several situations, where the system can detect that it actually misunderstood what the user said:

- If there is a contradiction between the slots and the intent which have been detected.
- If the score of the detected intent or slot is low.
- If the user reacts negatively, by saying for example “You misunderstood me” or “You’re wrong”.

When facing this kind of situation, the goal is not only to detect the problem and deal with it in the current dialogue, but is also to infer/obtain new information allowing the system to adapt itself.

[4] See https://lihlith.limsi.fr/dialog.php. To try it, use the following login and password: “lihlith” and “recipe?”.

<table>
<tr><th></th><th>Utterance</th><th>System action</th></tr>
<tr><td>User</td><td>i'd like to prepare a mousse</td><td>The system hasn't detected that "mousse" corresponds to the slot "recipe" but has detected the intent "REC"</td></tr>
<tr><td>System</td><td>Are you asking for a specific recipe?</td><td>Ask confirmation about the intent detected</td></tr>
<tr><td>User</td><td>Yes</td><td></td></tr>
<tr><td>System</td><td>Can you tell me the name of the recipe alone?</td><td>Ask for the missing slot</td></tr>
<tr><td>User</td><td>Mousse</td><td>Save the annotated utterance:<table><tr><td>i'd</td><td>like</td><td>to</td><td>prepare</td><td>a</td><td>mousse</td><td>INTENT</td></tr><tr><td>O</td><td>O</td><td>O</td><td>O</td><td>O</td><td>b-recipe</td><td>REC</td></tr></table></td></tr>
<tr><td>System</td><td>I found 3 recipes, including: atkins-friendly chocolate mousse, chocolate mousse cake (vegan) and chocolate mousse .
Which one do you want?</td><td>Answer the initial query</td></tr>
</table>

Fig. 1 Example of contradiction between detected slot and intent

We consider the example presented in Fig. 1, where the system should detect a contradiction between a slot and an intent.[5]

When facing this kind of problem, the system could first ask the user to clarify the slots and the intent it has detected. This clarification process would thus lead to the creation of an annotated user's utterance.

In order to allow the system to learn from this clarification, and to be able to reuse it directly or in future conversations, this annotated data can first be saved in a specific file. Then, when detecting a misunderstanding, the system would, before even asking the user about clarification, look into this specific file if the user's utterance that the system doesn't understand, corresponds or has high similarities with an annotated utterance in the file.

Moreover, we could consider retraining our NLU model with the accumulated annotated user's utterances consisting in new training examples. This could be done for example after a specified number of annotated user's utterances have been added to this file and if the annotated data seems to be useful and reliable according to user's feedbacks. However, we should be careful about respecting the proportions of the different slots and intents in the training data, when retraining the model with the new examples, in order to minimize the problem of catastrophic forgetting and performance decrease [9].

We could also infer a pattern from the annotated user's utterance and generate automatically annotated data from this pattern to retrain our understanding model. In the example presented in Fig. 1, we could infer the pattern "i'd like to prepare a $recipe".

One can also face a situation where the system detects a new intent as it is shown in Fig. 2.

[5]The intent REC means that the user searches for a recipe by giving the name of the recipe.

	Utterance	System action
User	i'd like to cook roast chicken	
System	OK, for roast chicken . Do you want the ingredients?	Begin sub-dialogue
User	What temperature for my oven?	
System	I don't know. Can you give me an example of an appropriate answer?	Ask for clue
User	210°C or 370 degree F	
System	I found "Preheat your oven to 200°C (390°F)" Are you satisfied with this answer?	Look in the recipe document for the sentence which is the most similar to the example of the user and ask for feedback
User	Yes	

Fig. 2 Example of new intent (sub-dialogue)

In such a situation, the question is not understood at all by the system and it then decides to ask the user to give an example of what could be a plausible answer to this question. Using this plausible answer, the system can apply other techniques to find the relevant piece of information, by searching in its database or in external resources (here the unstructured data it has access to). If the user confirms that the answer of the system is appropriate, the system could enter to another process to complete automatically its database. As proposed in [10], the system could thus check in its database if this new intent could be applied to other recipes. In the example of Fig. 2, this process could result in the creation of the feature "oven_temperature" and a value could be associated with the new feature for all the recipes needing an oven.

It is important that the system chooses the right moment to ask its question to the user. As it is done in [4], methods based on (inverse) reinforcement learning framework might be very interesting.

4.2.2 Data Enrichment

This section presents the second way to perform LL in order to improve our system, which is about new factual knowledge acquisition and more specifically enriching the recipe database.

At some point in the dialogue, the user and the system are speaking about one specific recipe. We consider, in this case, that if a piece of information is missing in the database, like the time needed, the difficulty or the serving, the system should ask the user about it, in oder to enrich its database.

This situation is presented in Fig. 3.

RL could be used to train the system to decide when it should ask for a new piece of information as it is done in [4]. RL will also allow the system to learn which piece of information is the most interesting to ask. The system can base its decisions on

	Utterance	System action
System	Hello, are you hungry?	
User	i'd like to cook basic scones	
System	You're telling me about basic scones . Is that OK?	Ask for confirmation
User	Ok	
System	OK, for basic scones . Do you want the ingredients?	
User	Yes	
System	The ingredients are: 1 cup flour (100g), 1 tsp baking powder (5g),1 pinch salt, 2 Tbs butter (50g), 1 Tbs sugar (50g),1 cup milk (125ml) + extra milk for glazing 2. Have you ever cooked basic scones?	The system has detected that at least a piece of information about the recipe is missing in the DB and decides to interact by asking how familiar the user is with the recipe.
User	Yes	
System	Do you remember how much time did it take?	Ask the user for the missing data
User	I needed approximately 30 min	Add to the database "cooking_time = 30 min"
System	OK, thank you very much!	

Fig. 3 Example of data enrichment

various parameters such as the number of time such a missing piece of information is asked by different users, if the current user seems to be willing to answer system's questions, if he seems to be knowledgeable in cooking, etc. And obviously RL will help to build a model on such an action to collect new data in a proactive way.

5 Conclusion and Discussion

In this paper we try to define what a LL task-oriented dialogue system could be and to apply this definition to our existing chatbot devoted to the cooking domain. We consider that two ways of improvement are possible for our chatbot: the system can learn to better understand and can enrich its database. We consider different situations in which the system might want to interact with the user in order to obtain his help. In such cases, a RL framework can be used to learn in which condition a question could be asked.

When applying our hypotheses to our chatbot, we can observe that a lifelong learning task-oriented dialogue system involves:

- having for the system, the ability to understand from user's utterance when it can learn new knowledge and the ability to handle it in the current context;
- some criteria or score functions which evolve over time to know when to ask questions and what questions to ask and to whom;

- a way to store the learned knowledge;
- a criterion to know when to retrain a model and an other one to know when to permanently store the learned knowledge in case of knowledge base enrichment.

However, we presented only simple LL aspects and put aside the complex ones such as LL for improving dialogue quality. Such an aspect would be interesting in any case and more specifically when working with conversational systems.

Acknowledgements This work has been partially funded by the LIHLITH project (ANR-17-CHR2-0001-03), and supported by ERA-Net CHIST-ERA, and the "Agence Nationale pour la Recherche" (ANR, France).

References

1. Chandramohan S, Geist M, Lefevre F, Pietquin O (2011) User simulation in dialogue systems using inverse reinforcement learning. In: Twelfth annual conference of the international speech communication association
2. Chen Z, Liu B (2016) Lifelong machine learning. Morgan & Claypool Publishers, San Rafael
3. Chen Z, Liu B, Brachman R, Stone P, Rossi F (2018) Lifelong machine learning: Second Edition. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool Publishers, San Rafael. https://books.google.fr/books?id=JQ5pDwAAQBAJ
4. Li J, Miller AH, Chopra S, Ranzato M, Weston J (2017) Learning through dialogue interactions by asking questions. In: ICLR. https://arxiv.org/pdf/1612.04936
5. Maltoni D, Lomonaco V (2018) Continuous learning in single-incremental-task scenarios. arXiv:180608568
6. Mazumder S, Ma N, Liu B (2018) Towards a Continuous Knowledge Learning Engine for Chatbots. arXiv:1802.06024 eprint1802.06024
7. Neuraz A, Campillos Llanos L, Burgun A, Rosset S (2018) Natural language understanding for task oriented dialog in the biomedical domain in a low ressources context, NIPS Workshop. In: Machine Learning for Health (ML4H): Moving beyond supervised learning in healthcare, Montréal, Québec, Canada
8. Papangelis A, Karkaletsis V, Makedon F (2012) Evaluation of online dialogue policy learning techniques. In: Proceedings of the eighth international conference on language resources and evaluation (LREC-2012), European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2012/pdf/291_Paper.pdf
9. Parisi GI, Kemker R, Part JL, Kanan C, Wermter S (2018) Continual Lifelong Learning with Neural Networks: a review. arXiv:1802.07569, eprint1802.07569
10. Peñas A, Veron M, Pradel C, Otegi A, Echegoyen G, Rodrigo A (2019) Continuous Learning for Question Answering. In: Tenth international workshop on spoken dialogue systems technology (IWSDS)
11. Thrun S, Mitchell TM (1995) Lifelong robot learning. Robot Autono Syst 15:25–46
12. Tur G (2011) Spoken language understanding: systems for extracting semantic information from speech. Wiley, New york. https://www.microsoft.com/en-us/research/publication/spoken-language-understanding-systems-for-extracting-semantic-information-from-speech/
13. Young S (2006) Using pomdps for dialog management. In: Spoken language technology workshop, 2006. IEEE, pp 8–13. http://mi.eng.cam.ac.uk/research/dialogue/slt06_sjy-talk.pdf

Towards Understanding Lifelong Learning for Dialogue Systems

Mark Cieliebak, Olivier Galibert, and Jan Deriu

Abstract Lifelong learning is the ability of a software system to adapt to new situations during its lifetime. We explore how this paradigm can be applied to dialogue systems, how it might be implemented, and how we can evaluate the lifelong learning progress.

1 Introduction

Chatbots, dialogue systems, conversational user interfaces–the names may differ, but the basic idea is the same: "intelligent" computer systems that can interact with humans in natural language. These systems have become more and more popular in the past years, and there is an increasing interest in spoken and written dialogue systems in research and industry. Prominent examples include automatic customer support agents, smart home devices such as Amazon Alexa or Apple's Siri, and in-car operating systems. While implementing a successful and reliable dialogue systems is already a challenge, "lifelong learning" even adds an additional twist: the system should be able to adapt to new situations during its lifetime. More precisely, the dialogue system learns to handle new situations by interacting with its environment (e.g. asking a domain expert, scraping the web), instead of being retrained by a machine learning expert. For instance, a chatbot for travel advice might be confronted with a new location that is not yet in its knowledge base. One strategy to deal with this situation could be to ask the user to give additional information (e.g. in which

M. Cieliebak (✉) · J. Deriu
Zurich University of Applied Sciences (ZHAW), Winterthur, Switzerland
e-mail: ciel@zhaw.ch

J. Deriu
e-mail: deri@zhaw.ch

O. Galibert
Laboratoire national de métrologie et d'essais, Trappes, France
e-mail: Olivier.Galibert@lne.fr

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_33

country, GPS coordinates etc.), then explore the web to find information about the location (e.g. databases, Wikipedia or travel reviews), and finally analyze, structure, and integrate the information into the chatbots' knowledge base.

In this paper, we attempt to make a step forward towards understanding what lifelong learning in the context of dialogue systems means. In order to achieve this, we first briefly introduce both concepts independently and discuss typical settings and applications. Then we describe the the impact of applying lifelong learning to dialogue systems (in Sect. 4). Finally, we turn to the important question how we could measure the success (or failure) of lifelong learning in the context of dialogue systems (Sect. 5).

2 What Is a Dialogue System?

In the following, we introduce the concept of a dialogue system. *A dialogue system allows the user to converse with a computer system using natural language*. Such systems can be applied to a variety of tasks, e.g.:

- *Virtual Assistants*, which are developed to aid its users in every-day tasks, such as scheduling appointments. They usually operate on predefined actions, which can be triggered by voice command.
- *Interaction with Information Systems*, by asking questions or finding a piece of information (e.g. the most suitable hotel in town).
- *Training environments*, where the dialogue systems are developed to train students in the interaction with medical patients or train military personnel in questioning a witness.
- *Answering Questions*, where the dialogue system can answer specific questions of a user. These might be factoid questions or more complex questions.

Dialogue systems usually structure dialogues in *turns*, each turn is defined by one or more *utterances* from one speaker. Two consecutive turns between two different speakers is called an *exchange*. Multiple exchanges constitute a *dialogue*. Another correlated view, is to interpret each turn or each utterance as an action. The main component of a dialogue system is the dialogue strategy, which defines the content of the next utterance and thus the behaviour of the dialogue system. There are many different approaches to design a dialogue strategy, which are partly dictated by the application of the dialogue system. However, there are three broad classes of dialogue systems, which we encounter in the literature: task-oriented systems, conversational agents and interactive question answering systems.[1] We identified the following characteristic features, which help differentiate between the three different classes: is the system developed to solve a task, does the dialogue follow a structure, is the

[1]In recent literature, the distinction is made only between the first two classes of dialogue systems [1, 4, 7]. However, interactive question answering systems cannot be completely placed in either of the two categories.

domain restricted or is it open domain, does the dialogue span over multiple turns, are the dialogues rather long or efficient, who takes the initiative, and what is the interface used (text, speech, multi-modal).

- *Task-oriented systems* are developed to help the user solve a specific task as efficiently as possible. The dialogues are characterized by following a clearly defined structure, which is derived from the domain. The dialogues follow mixed initiative: both, the user and the system can take the lead. Usually, the systems found in the literature are built for speech input and output. However, task-oriented systems in the domain of assistance are built on multi-modal input and output.
- *Conversational agents* display a more unstructured conversation, as their purpose is to have open-domain dialogues with no specific task to solve. Most of these systems are built to emulate social interactions and thus longer dialogues are desired.
- *Question Answering (QA) systems* are built for the specific task of answering questions. The dialogues are not defined by a structure as with task-oriented systems, however they mostly follow the question and answer style pattern. QA systems may be built for a specific domain, but also be tilted towards more open domain questions. Usually, the domain is dictated by the underlying data, e.g. knowledge bases or text snippets from forums. Traditional QA systems work on a singe-turn interaction, however, there exist systems that allow multiple turns to cover follow-up questions. The initiative is mostly done by the user who asks questions.

3 What Is Lifelong Learning?

In the most abstract way, *Lifelong Learning (LL) is the ability of a system to use past experiences to adapt to future challenge*. For the purpose of this paper, we exploit the definition of LL from Chen and Liu [2], which we summarize in the following:

Lifelong learning is a continuous learning process. Given that the learner has learned N tasks. When faced with the $(N + 1)$th task the learner leverages past knowledge to help learn the new task. The goal is to optimize on both the new task and the previous tasks. The three components are: continuous learning, knowledge accumulation and maintenance and leverage past knowledge to learn new tasks. There are some additional considerations to be made considering the above definition.

- The learner learns new tasks continuously, however, in contrast to transfer-learning, the learner improves or at least does not deteriorate its performance on the old tasks. Ideally, by learning new tasks, the performance on the previous tasks improves.
- The learner is not restricted to a certain task or domain. On the contrary, the learner is encouraged to learn different types of tasks (e.g. sentiment analysis, named entity recognition, etc.) and domains.
- Ideally, the learner is *self-motivated* and able to find its own learning tasks and data by interacting with the environment.

Note that this definition is strongly focused on “knowledge improvement”, whereas in the setting of LL for dialogue systems, there are also other goals, as we discuss in the following section. In addition, we would like to mention that the concept of a “task” may cover situations of varying complexity, ranging from single new instances (e.g. a new person in a face recognition system) up to new domains (e.g. switching from cooking to car tuning for a questions answering systems). Finally, note that several other terms have been coined and used for very similar learning paradigms of systems that improve over time, such as continuous learning [3, 6], meta-learning [9], active learning [8], or transfer learning [5]. For a more elaborate introduction of LL, see the recent book by Chen and Liu [2], which gives a good overview of LL in general and describes applications in various fields.

4 Lifelong Learning for Dialogue Systems

While the definition of lifelong learning given by [2] is very general, we attempt now to apply the definition to dialogue systems. These systems allow its users to converse with a computer via natural language. This implies a high level of interactivity. Thus, the focus of applying LL to dialogue systems should lie in the interactive nature of the dialogue. Furthermore, LL describes the capability of the dialogue system to learn to handle new situations throughout its deployment, i.e. without being re-trained by a machine-learning expert. Ideally, the learning takes place in a self-driven and autonomous manner. This does not exclude (it rather encourages) the assistance of a “domain expert”, i.e. a type of user who takes the role of a teacher.

We assume that the dialogue system is an agent that interacts with its environment. The environment includes humans as well as having access to structured and unstructured knowledge sources (e.g. knowledge bases, Wikipedia, Twitter). When faced with a new situation, the dialogue system has to learn how to handle this new situation. This does not necessarily means that the dialogue system directly adapts to the new situations. Rather, through interaction with its environment, the dialogue system learns to handle the situations over time.

There are various aspects to a dialogue system which can be improved over time:

- *Language Understanding*: Here, the dialogue systems’ capability of parsing the user input is the focal point. This is the case, for instance, when new request types occur over time, for instance if a system was only faced with simple factual questions until now, and the system is suddenly confronted with complex questions.
- *Dialogue Behaviour* is concerned with the “soft” quality factors of a dialogue, such as human-likeness, appropriateness of responses, efficiency of reaching a goal, engaging utterances etc. Typically, the DS asks after a user interaction for feedback, which is then used to improve the behaviour over time. Thus, the DS leverages past experiences to improve its future behaviour. Note that in this case, there are no external catalysts that trigger LL, but there is an “intrinsic motivation” of the system itself.

- *Knowledge Induction* is concerned with accumulating more information. This means adapting the knowledge base (KB) with new or updated knowledge, which can be factual knowledge or unstructured. Here, new situations are in the context of handling new entities and relations which are not in the current knowledge base.
- *Capability Improvement* is concerned with extending the functionality of the DS. This can range from domain adaption (e.g. moving from asian recipies to the pasta domain) up to integrating new skills (e.g. reporting weather forecasts for a personal assistant)

In each case the dialogue system needs to improve its aspects over time. Each time it is faced with a new situation one or more of the aforementioned aspects need to be adapted. In the context of dialogue systems, this adoption can be done by means of interacting with a "domain" expert. More precisely, the goal is to remove the need to rely on a dialogue system expert who would retrain the different components of the dialogue system. Rather the domain-expert teaches the dialogue system how to handle a new situation through interaction. Note that in some cases the system may be able to learn how to handle the situation autonomously, especially in the case where it can aggregate data from some external sources. Thus, a LL enhanced dialogue system is able to learn to adapt to new situations by interacting with its environment and not by means of retraining components.

5 Evaluation of Lifelong Learning for Dialogue Systems

The above definition of LL for dialogue systems sets a strong focus on learning to handle new situations by interacting with its environment. Thus, the LL component of the dialogue system needs to be trained and evaluated with this in mind. More precisely, the interaction with an environment lies at the centre of the training and evaluation. The environment should enable the interaction the dialogue system will encounter during deployment. This includes the interaction with a domain expert.

In general, LL evaluation methods need to be reproducible in order to measure improvements and changes over time. One straightforward way of doing this is to deploy a dialogue system and let humans interact with it. However, this is very time consuming and expensive, and alternatives with less or no humans in the loop are desired. One major issue that is particular for evaluating dialogue systems is that they produce their "result"—the dialogue—during the interaction with their environment. Thus, any automated environment environment has to provide artificial users, and building them can be as complex as building the dialogue system itself.

When it comes to LL evaluation, additional complexity arises due to the fact that the interaction with the expert needs to be simulated as well. For instance, the dialogue system may ask an expert for advice about a new entity or topic. In general, the evaluation system cannot know in advance *which* questions the dialogue system will ask—hence, it is hard to simulate.

Experimental Evaluation Environment We are currently working on an experimental setting to automate the evaluation of knowledge acquisition and capability improvement. We work in the cooking domain, where the dialogue system is developed to assist the user by answering questions about cooking, e.g. providing recipes, giving advice or providing tutorials. Typical question might be "How do i prepare linguini?", which is answered with a corresponding recipe from the database.

In order to evaluate the LL capabilities of the dialogue system, we deploy it in a simulated environment, which consists of:

- Evaluation agent: provides the questions and evaluates the answers given by the dialogue system. The agent institutes new situations by asking about entities which were not present in any training set of the dialogue system (e.g. enchilada), by asking types of questions which the dialogue system did not encounter yet (e.g. "How do i clean my oven?"), or by asking questions about unseen domains (e.g. Chinese food).
- Expert: provides advice to the dialogue system when stuck. The dialogue system can ask clarifying questions to the expert before it tries to answer the question of the evaluation agent. However, this comes at a cost, i.e. each interaction with the expert has its fee, and thus, the system should learn to efficiently interact with the expert.

  We envision that the dialogue system asks questions from a list of predefined templates, which the (automated) expert can easily parse and answer. These are, for instance, "What is <X>?" or "Is <X> a relevant entity for this question?".

  The domain expert has at its hand a large collection of pre-recorded dialogues on the domain, and returns extracts of these dialogue that match to the clarification question.

The evaluation measures the capability of the LL component to adapt to the new situations. This capability is measures by the number of interactions needed with the expert system before answering the initial question correctly. A system with a strong LL component should adapt to new situations quickly.

6 Conclusion

Implementing lifelong learning for a dialogue system may aim at 1. extending the underlying knowledge base (Knowledge Induction); 2. handling more complex user interactions (Language Understanding); 3. improving the perceived quality of the resulting dialogues (Dialogue Behaviour); or 4. extending the functionality of the system (Capability Improvement) over time.

While stating these goals is simple, implementing a system that achieves any of these four goal is far from trivial. To the best of our knowledge, most approaches in research currently tackle the first dimension (Knowledge Induction), while there is almost no solution (yet) for the other three.

One important challenge is to evaluation the progress of LL in such systems. In order to avoid time-consuming and costly human evaluations, automated environments are required. We are currently working on such a system, which shall be presented as shared task in 2020.

Acknowledgements This paper has been partially funded by the LIHLITH project, supported by ERA-NET CHIST-ERA and Swiss National Science Foundation (20CH21_174237).

References

1. Chen H, Liu X, Yin D, Tang J (2017) A survey on dialogue systems: recent advances and new frontiers. SIGKDD Explor Newsl 19(2):25–35. https://doi.org/10.1145/3166054.3166058
2. Chen Z, Liu B (2016) Lifelong machine learning. Morgan & Claypool Publishers, San Rafael
3. Goodfellow IJ, Mirza M, Xiao D, Courville A, Bengio Y (2013) An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv:13126211
4. Jurafsky D, Martin JH (2017) Speech and Language Processing, draft of 3rd edition (Chap: Dialog Systems and Chatbots)
5. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359. https://doi.org/10.1109/TKDE.2009.191
6. Parisi GI, Kemker R, Part JL, Kanan C, Wermter S (2018) Continual lifelong learning with neural networks: a review. arXiv:180207569
7. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2017) A survey of available corpora for building data-driven dialogue systems. arXiv:151205742
8. Settles B (2012) Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, vol 6, no 1, pp 1–114
9. Vilalta R, Drissi Y (2002) A perspective view and survey of meta-learning. Artif Intell Rev 18(2):77–95

Question Answering and Other Dialogue Applications

Incremental Improvement of a Question Answering System by Re-ranking Answer Candidates Using Machine Learning

Michael Barz and Daniel Sonntag

Abstract We implement a method for re-ranking top-10 results of a state-of-the-art question answering (QA) system. The goal of our re-ranking approach is to improve the answer selection given the user question and the top-10 candidates. We focus on improving deployed QA systems that do not allow re-training or when re-training comes at a high cost. Our re-ranking approach learns a similarity function using n-gram based features using the query, the answer and the initial system confidence as input. Our contributions are: (1) we generate a QA training corpus starting from 877 answers from the customer care domain of T-Mobile Austria, (2) we implement a state-of-the-art QA pipeline using neural sentence embeddings that encode queries in the same space than the answer index, and (3) we evaluate the QA pipeline and our re-ranking approach using a separately provided test set. The test set can be considered to be available after deployment of the system, e.g., based on feedback of users. Our results show that the system performance, in terms of top-n accuracy and the mean reciprocal rank, benefits from re-ranking using gradient boosted regression trees. On average, the mean reciprocal rank improves by 9.15%.

1 Introduction

In this work, we examine the problem of incrementally improving deployed QA systems in an industrial setting. We consider the domain of customer care of a wireless network provider and focus on answering frequent questions (focussing on the

M. Barz (✉) · D. Sonntag
German Research Center for Artificial Intelligence, Saarland Informatics Campus, D3 2, 66123 Saarbrücken, Germany
e-mail: michael.barz@dfki.de

D. Sonntag
e-mail: daniel.sonntag@dfki.de

M. Barz
Saarbrücken Graduate School of Computer Science, Saarbrücken, Germany

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_34

long tail of the question distribution [2]). In this setting, the most frequent topics are covered by a separate industry-standard chatbot based on hand-crafted rules by dialogue engineers. Our proposed process is based on the augmented cross-industry standard process for data mining [23] (augmented CRISP data mining cycle). In particular, we are interested in methods for improving a model after its deployment through re-ranking of the initial ranking results. In advance, we follow the steps of the CRISP cycle towards deployment for generating a state-of-the-art baseline QA model. First, we examine existing data (*data understanding*) and prepare a corpus for training (*data preparation*). Second, we implement and train a QA pipeline using state-of-the-art open source components (*modelling*). We perform an evaluation using different amounts of data and different pipeline configurations (*evaluation*), also to understand the nature of the data and the application (*business understanding*). Third, we investigate the effectiveness and efficiency of re-ranking in improving our QA pipeline after the *deployment* phase of CRISP. Adaptivity after deployment is modelled as *(automatic) operationalisation* step with external reflection based on, e.g., user feedback. This could be replaced by introspective meta-models that allow the system to enhance itself by metacognition [23]. The QA system and the re-ranking approach are evaluated using a separate test set that maps actual user queries from a chat-log to answers of the QA corpus. Sample queries from the evaluation set with one correct and one incorrect sample are shown in Table 1.

With this work, we want to answer the question whether a deployed QA system that is difficult to adapt and that provides a top-10 ranking of answer candidates, can be improved by an additional re-ranking step that corresponds to the operationalisation step of the augmented CRISP cycle. It is also important to know the potential gain and the limitations of such a method that works on top of an existing system. We hypothesise that our proposed re-ranking approach can effectively improve ranking-based QA systems.

2 Related Work

The broad field of QA includes research ranging from retrieval-based [8, 9, 15, 29] to generative [20, 21], as well as, from closed-domain [10, 16] to open-domain QA [7, 13, 18, 20]. We focus on the notion of improving an already deployed system.

For QA dialogues based on structured knowledge representations, this can be achieved by maintaining and adapting the knowledgebase [24–26]. In addition, [23] proposes metacognition models for building self-reflective and adaptive AI systems, e.g., dialogue systems, that improve by introspection. Buck et al. present a method for reformulating user questions: their method automatically adapts user queries with the goal to improve the answer selection of an existing QA model [5].

Other works suggest humans-in-the-loop for improving QA systems. Savenkov and Agichtein use crowdsourcing for re-ranking retrieved answer candidates in a real-time QA framework [19]. In Guardian, crowdworkers prepare a dialogue system based on a certain web API and, after deployment, manage actual conversations with

Table 1 Sample queries with a correct and an incorrect answer option according to our test set. We report the answers' rank of the baseline model that we used for our re-ranking experiments

User query	Correct answer	Incorrect answer
Bekomme ich bei Vertragsverlängerung ein neues Handy? (*Do I get a new phone when extending my contract?*)	Ab wann Sie Ihre Rufnummern verlängern können und welche Angebote bei einer Vertragsverlängerung auf Sie warten, sehen Sie in Ihrem persönlichen Kundenportal Mein T-Mobile […] (*In your online customer area of T-Mobile, you can see when you can continue your telephone numbers and which offers await you after extending your contract […]*) **(rank 1)**	Suchen Sie ein neues Gerät, das genau Ihre Bedürfnisse und Anforderungen erfüllt? Sie wollen rechtzeitig über Neuerungen informiert werden? […] (*You are looking for a new device that satisfies all your requirements? You want to get recent news? […]*) **(rank 5)**
tarife ohne bi**m**dung (*plans without bi**m**ding contract*) – `misspelled`	Wenn Sie bereits ein Handy besitzen und nur eine Simkarte benötigen, haben wir genau das Richtige für Sie: die Klax-SIM. […] (*If you own a new phone and all you need is a SIM card, we got exactly the right offer: the Klax-SIM. […]*) **(rank 3)**	Eine Übersicht über unsere aktuellen My Mobile Handytarife inklusive aller wichtigen Details finden Sie auf der Tarifseite. […] (*An overview of our current service plans with all important details can be found on our website. […]*) **(rank 1)**
Kosten für vertragübernahme (*costs for a contract transfer*)	Sie können Verträge an andere Personen übergeben, zusammenlegen oder trennen. Die Kosten belaufen sich auf […] Ausführliche Informationen zum Thema finden Sie in den FAQ. (*You can transfer, join and split contracts from and to other persons. The costs are […] More detailed information can be found in our FAQ.*) **(rank 10)**	Ein Zukauf von Freiminuten ist nicht möglich und bei unseren aktuellen Tarifen auch nicht notwendig, da Freiminuten hier unlimitiert sind. (*You cannot buy additional minutes. However, that's not required with our plans, because minutes are unlimited.*) **(rank 1)**
Kreditkarte (*credit card*)	Eine Zahlung mittels Kreditkarte ist selbstverständlich bei uns möglich. Sollten Sie Ihre Zahlungsart auf Kreditkarte ändern oder Ihre Daten aktualisieren wollen, können Sie dies direkt über unseren LiveChat veranlassen. […] (*Of course, you can pay with your credit card. If you want to change your payment settings to credit card or if you want to update your data, you can do so using our LiveChat. […]*) **(rank 7)**	Die Änderung Ihrer Kreditkartendaten ist zu Ihrer Sicherheit nur telefonisch bei der Serviceline unter […] (*For security reasons, you can change your credit card data via phone using our service hotline at […] only*) **(rank 1)**
Hallo, ich möchte ein iPhone 7 kaufen (Ratenzahlung). Ich hab schon ein Vertrag (bis 09.2017)..wenn ich das verlängern möchte muss ich die Raten von meine altes Handy weiter zahlen? Lg (*Hello, I'd like to buy an iPhone 7 (paying by instalments). I have a contract (till 09/2017)..if I want to extend it, do I need to pay the remaining rates for my old phone? Kind regards*)	Ratenzahlungen oder Stundungen bei offenen Rechnungsbeträgen bietet T-Mobile NICHT an […] (*T-Mobile does NOT offer payment by instalments or deferred payments for outstanding bill amounts.*) **(not in top-10)**	Mit der Umstellung auf LTE hat sich nichts am Geschwindigkeitsprofil inklusive der erreichbaren Maximalgeschwindigkeiten Ihres aktuellen Tarifes geändert. […] (*The transition to LTE (4G) does not affect the maximum data transfer rate of your present service plan.*) **(rank 1)**

users [12]. EVORUS learns to select answers from multiple chatbots via crowdsourcing [11]. The result is a chatbot ensemble excels the performance of each individual chatbot. Williams et al. present a dialogue architecture that continuously learns from user interaction and feedback [27].

We propose a re-ranking algorithm similar to [19]: we train a similarity model using n-gram based features of QA pairs for improving the answer selection of a retrieval-based QA system.

3 Question Answering System

We implement our question answering system using state-of-the-art open source components. Our pipeline is based on the Rasa natural language understanding (NLU) framework [4] which offers two standard pipelines for text classification: *spacy_sklearn* and *tensorflow_embedding*. The main difference is that *spacy_sklearn* uses Spacy[1] for feature extraction with pre-trained word embedding models and Scikit-learn [17] for text classification. In contrast, the *tensorflow_embedding* pipeline trains custom word embeddings for text similarity estimation using TensorFlow [1] as machine learning backend. Figure 1 shows the general structure of both pipelines. We train QA models using both pipelines with the pre-defined set of hyper-parameters. For *tensorflow_embedding*, we additionally monitor changes in system performance using different epoch configurations.[2] Further, we compare the performances of pipelines with or without a spellchecker and investigate whether model training benefits from additional user examples by training models with the three different versions of our training corpus including no additional samples (`kw`), samples from 1 user (`kw+1u`) or samples from 2 users (`kw+2u`) (see section Corpora). All training conditions are summarized in Table 2. Next, we describe the implementation details of our QA system as shown in Fig. 1: the spellchecker module, the subsequent pre-processing and feature encoding, and the text classification. We include descriptions for both pipelines.

Spellchecker. We address the problem of frequent spelling mistakes in user queries by implementing an automated spell-checking and correction module. It is based on a Python port[3] of the SymSpell algorithm[4] initialized with word frequencies for German.[5] We apply the spellchecker as first component in our pipeline.

Pre-Processing and Feature Encoding. The *spacy_sklearn* pipeline uses Spacy for pre-processing and feature encoding. Pre-processing includes the generation of a Spacy `document` and tokenization using their German language model

[1] https://spacy.io/.

[2] https://rasa.com/docs/nlu/components/#intent-classifier-tensorflow-embedding.

[3] https://github.com/mammothb/symspellpy.

[4] https://github.com/wolfgarbe/SymSpell.

[5] German 50k: https://github.com/hermitdave/FrequencyWords.

Table 2 Considered configurations for QA pipeline training

	spacy_sklearn	*tensorflow_embedding*
Parameters	Default	Default with `epochs` $\in \{10, 50, 100, 300, 600\}$
Spellchecking	`yes`, `no`	
Training corpus	`kw`, `kw+1u`, `kw+2u`	

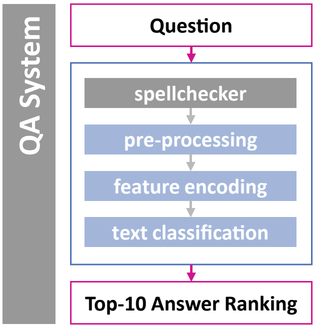


Fig. 1 The basic configuration of the QA pipeline, which is a part of our complete QA system architecture with the re-ranking algorithm

`de_core_news_sm` (v2.0.0). The feature encoding is obtained via the `vector` function of the Spacy document that returns the mean word embedding of all tokens in a query. For German, Spacy provides only a simple dense encoding of queries (no proper word embedding model).

The pre-processing step of the *tensorflow_embedding* pipeline uses a simple whitespace tokenizer for token extraction. The tokens are used for the feature encoding step that is based on Scikit-learn's `CountVectorizer`. It returns a bag of words histogram with words being the tokens (1-grams).

Text Classification. The *spacy_sklearn* pipeline relies on Scikit-learn for text classification using a support vector classifier (SVC). The model confidences are used for ranking all answer candidates; the top-10 results are returned.

Text classification for *tensorflow_embedding* is done using TensorFlow with an implementation of the StarSpace algorithm [28]. This component learns (and later applies) one embedding model for user queries and one for the answer id. It minimizes the distance between embeddings of QA training samples. The distances between a query and all answer ids are used for ranking.

3.1 Corpora

In this work, we include two corpora: one for training the baseline system and another for evaluating the performance of the QA pipeline and our re-ranking approach. In the following, we describe the creation of the training corpus and the structure of the test corpus. Both corpora have been anonymised.

Training Corpus. The customer care department provides 877 answers to common user questions. Each answer is tagged with a variable amount of keywords or key-phrases ($M = 3.81$, $SD = 5.92$), 3338 in total. We asked students to augment the training corpus with, in total, two additional natural example queries. This process can be scaled by crowdsourcing for an application in productive systems that might include more answers or that requires more sample question per answer or both. The full dataset contains, on average, 5.81 sample queries per answer totalling in 5092 queries overall. For model training, all questions (including keywords) are used as input with the corresponding answer as output. We generated three versions of the training corpus: keywords only (`kw`, $n = 3338$), keywords with samples from 1 user (`kw+1u`, $n = 4215$) and keywords with samples from 2 users (`kw+2u`, $n = 5092$).

Evaluation Corpus. The performance of the implemented QA system and of our re-ranking approach is assessed using a separate test corpus. It includes 3084 real user requests from a chat-log of T-Mobile Austria, which are assigned to suitable answers from the training corpus (at most three). The assignment was performed manually by domain experts of the wireless network provider. We use this corpus for estimating the baseline performance of the QA pipeline using different pipeline configurations and different versions of the training corpus. In addition, we use the corpus for evaluating our re-ranking approach per cross-validation: we regard the expert annotations as offline human feedback. The queries in this corpus contain a lot of spelling mistakes. We address this in our QA pipeline generation by implementing a custom spell-checking component.

4 Baseline Performance Evaluation

We evaluate the baseline model using all training configurations in Table 2 to find a well-performing baseline for our re-ranking experiment. We use the evaluation corpus as reference data and report the top-1 to top-10 accuracies and the mean reciprocal rank for the top-10 results (MRR@10) as performance metrics. For computing the top-n accuracy, we count all queries for which the QA pipeline contains a correct answer on rank 1 to n and divide the result by the number of test queries. The MRR is computed as the mean of reciprocal ranks over all test queries. The reciprocal rank for one query is defined as $RR = \frac{1}{rank}$: The RR is 1 if the correct answer is ranked first, 0.5 if it is at the second rank and so on. We set RR to zero, if the answer is not contained in the top-10 results.

Results. Figure 2 shows the accuracy and MRR values for all conditions. We only restrict *tensorflow_embedding* to the default number of `epochs` which is 300. At the corpus level, we can observe that the accuracy and the MRR increase when training with additional user annotations for all pipeline configurations. For example, the *spacy_sklearn* pipeline without spell-checking achieves a top-10 accuracy of 0.317 and a MRR of 0.139 when using the `kw` training corpus with keywords only. Both measures increase to 0.33 and 0.165, respectively, when adding two natural queries

Fig. 2 Performance metrics in terms of top-1 to top-10 accuracy and MRR@10 of both QA pipelines for different pipeline configurations and training corpora

pipeline	spell check	corpus	Accuracy (top-n) 1	2	3	4	5	6	7	8	9	10	MRR @10
spacy-sklearn	*no*	*kw*	0.073	0.129	0.169	0.199	0.225	0.250	0.270	0.288	0.305	0.317	0.139
		kw+1u	0.095	0.156	0.194	0.223	0.246	0.265	0.287	0.304	0.317	0.332	0.161
		kw+2u	0.099	0.158	0.202	0.226	0.253	0.274	0.290	0.304	0.320	0.330	0.165
	yes	*kw*	0.095	0.150	0.205	0.238	0.265	0.286	0.304	0.322	0.342	0.356	0.167
		kw+1u	0.117	0.186	0.241	0.280	0.313	0.333	0.355	0.368	0.387	0.404	0.198
		kw+2u	0.127	0.203	0.250	0.295	0.327	0.354	0.375	0.393	0.412	0.428	0.212
tensorflow embedding (300 epochs)	*no*	*kw*	0.125	0.192	0.240	0.279	0.308	0.332	0.345	0.351	0.360	0.364	0.198
		kw+1u	0.156	0.227	0.277	0.318	0.347	0.365	0.381	0.394	0.404	0.413	0.233
		kw+2u	0.195	0.273	0.328	0.360	0.383	0.404	0.418	0.434	0.443	0.454	0.274
	yes	*kw*	0.194	0.276	0.326	0.363	0.390	0.412	0.424	0.435	0.445	0.456	0.275
		kw+1u	0.190	0.277	0.330	0.366	0.401	0.419	0.432	0.442	0.454	0.464	0.276
		kw+2u	0.180	0.274	0.332	0.375	0.408	0.435	0.451	0.463	0.478	0.492	0.275

Fig. 3 Performance metrics in terms of top-1 to top-10 accuracy and MRR@10 for the *tensorflow_embedding* pipeline with spell-checking for different training corpora and a different number of training `epochs`

corpus	epochs	Accuracy (top-n) 1	2	3	4	5	6	7	8	9	10	MRR @10
kw	*10*	0.193	0.261	0.298	0.321	0.334	0.349	0.361	0.371	0.378	0.387	0.255
	50	0.196	0.263	0.302	0.324	0.342	0.354	0.366	0.377	0.387	0.396	0.258
	100	0.181	0.265	0.312	0.345	0.365	0.380	0.396	0.405	0.416	0.425	0.259
	300	0.194	0.276	0.326	0.363	0.390	0.412	0.424	0.435	0.445	0.456	0.275
	600	0.145	0.220	0.284	0.326	0.362	0.387	0.400	0.415	0.430	0.439	0.232
kw+1u	*10*	0.189	0.253	0.294	0.320	0.334	0.350	0.364	0.376	0.386	0.396	0.252
	50	0.204	0.292	0.336	0.364	0.384	0.400	0.413	0.421	0.431	0.441	0.281
	100	0.235	0.318	0.369	0.401	0.428	0.445	0.459	0.474	0.485	0.493	0.316
	300	0.190	0.277	0.330	0.366	0.401	0.419	0.432	0.442	0.454	0.464	0.276
	600	0.183	0.276	0.333	0.373	0.400	0.425	0.442	0.459	0.472	0.485	0.275
kw+2u	*10*	0.189	0.268	0.311	0.337	0.359	0.377	0.389	0.400	0.413	0.425	0.263
	50	0.213	0.287	0.331	0.363	0.387	0.405	0.424	0.440	0.451	0.463	0.288
	100	0.208	0.299	0.353	0.391	0.417	0.438	0.454	0.466	0.478	0.494	0.296
	300	0.180	0.274	0.332	0.375	0.408	0.435	0.451	0.463	0.478	0.492	0.275
	600	0.196	0.300	0.356	0.393	0.421	0.438	0.455	0.470	0.486	0.498	0.292

for training. In some cases, adding only 1 user query results in slightly better scores. However, the overall trend is that more user annotations yield better results.

In addition, we observe performance improvements for pipelines that use our spell-checking component when compared to the default pipelines that do not make use of it: The *spacy_sklearn* `kw+2u` condition performs 9.8% better, the *tensorflow_embedding* `kw+2u` condition performs 3.8% better, in terms of top-10 accuracy. We can observe similar improvements for the majority of included metrics. Similar to the differentiation by corpus, we can find cases where spell-checking reduces the performance for a particular measure, against the overall trend.

Overall, the *tensorflow_embedding* pipelines perform considerably better than the *spacy_sklearn* pipeline irrespective of the remaining parameter configuration: the best performing methods are achieved by the *tensorflow_embedding* pipeline with spell-checking. Figure 3 sheds more light on this particular setting. It provides performance measures for all corpora and for different number of `epochs` used for model training. Pipelines that use 300 `epochs` for training range among the best for all corpora. When adding more natural user annotations, using 100 `epochs` achieves similar or better scores, in particular concerning the top-10 accuracy and the MRR. Re-ranking the top-10 results can only improve the performance in QA, if the correct answer is among the top-10 results. Therefore, we use the *tensorflow_embedding* pipeline with spellchecking, 100 `epochs` and the full training corpus as baseline for evaluating the re-ranking approach.

5 Re-ranking Approach

Our re-ranking approach compares a user query with the top-10 results of the baseline QA system. In contrast to the initial ranking, our re-ranking takes the content of the answer candidates into account instead of encoding the user query only. Our algorithm compares the text of the recent user query to each result. We include the answer text and the confidence value of the baseline system for computing a similarity estimate. Finally, we re-rank the results by their similarity to the query (see Algorithm 1).

Algorithm 1: Re-Ranking Algorithm

Input: a user query q; the corresponding list of top-10 results R including an answer a and the baseline confidence c;
Output: an updated ranking R'

```
begin
    R' ⟵ [];
    foreach (c, a) ∈ R do
        c' ⟵ similarity(q, a, c);
        R'.append((c', a));
    // sort R' by confidences c', descending
    sort(R');
    return R'
```

We consider a data-driven similarity function that compares linguistic features of the user query and answer candidates and also takes into account the confidence of the baseline QA system. This similarity estimate shall enhance the baseline by using an extended data and feature space, but without neglecting the learned patterns of the baseline system. The possible improvement in top-1 accuracy is limited by the top-10 accuracy of the baseline system (49.4%), because our re-ranking cannot choose from the remaining answers. Figure 4 shows how the re-ranking model is connected to the deployed QA system: it requires access to its in- and outputs for the additional ranking step.

We consider the gradient boosted regression tree for learning a similarity function for re-ranking similar to [19]. The features for model training are extracted from pre-processed query-answer pairs. Pre-processing includes tokenization and stemming of query and answer and the extraction of uni-, bi- and tri-grams from both token sequences.[6] We include three distance metrics as feature: the Jaccard distance, the cosine similarity,[7] and the plain number of n-gram matches between n-grams of a query and an answer.

[6]We use default word tokenizer, Snowball stemmer and n-gram extraction of the nltk toolkit [3].

[7]We use the implementation for Jaccard distance and cosine similarity as found in the following Github gist: gaulinmp/similarity_example.ipynb.

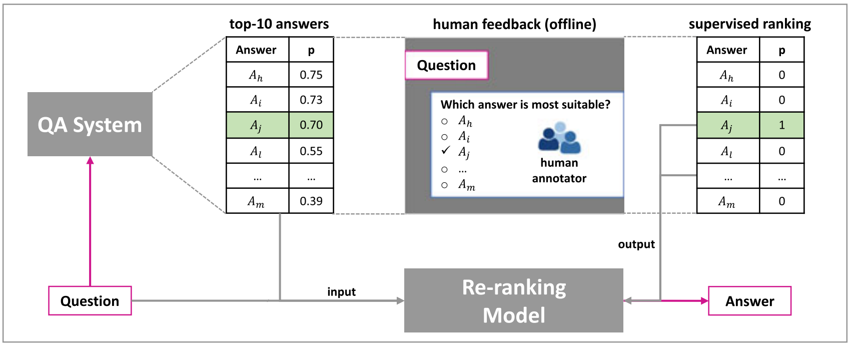


Fig. 4 Complete QA system architecture including the re-ranking model. The re-ranking model is trained using manually annotated data for generating a supervised/ideal ranking result for the top-10 answers from the QA system. Features are extracted from the user question and a particular answer candidate. At inference time, the re-ranking model is used to improve the initial top-10 ranking

6 Re-ranking Performance Evaluation

We compare our data-driven QA system with a version that re-ranks resulting top-10 candidates using the additional ranking model. We want to answer the question whether our re-ranking approach can improve the performance of the baseline QA pipeline after deployment. For that, we use the evaluation corpus ($n = 3084$) for training and evaluating our re-ranking method using 10-fold cross-validation, i.e., 90% of the data is used for training and 10% for testing with 10 different train-test splits.

The training and testing procedure per data split of the cross-validation is shown in Algorithm 2. For each sample query q in the train set C_{train}, we include the correct answer a^+ and one randomly selected negative answer candidate a^- for a balanced model training. We skip a sample, if the correct answer is not contained in the top-10 results: we include 49.4% of the data (see top-10 accuracy of the baseline QA model in Fig. 3). The baseline QA model r and the trained re-ranking method r' are applied to all sample queries in the test set C_{test}. Considered performance metrics are computed using the re-ranked top-10 $results$. We repeat the cross-validation 5 times to reduce effects introduced by the random selection of negative samples. We report the average metrics from 10 cross-validation folds and the 5 repetitions of the evaluation procedure.

Results. The averaged cross-validation results of our evaluation, in terms of top-n accuracies and the MRR@10, are shown in Table 3: the top-1 to top-9 accuracies improve consistently. The relative improvement decreases from 14.83% for the top-1 accuracy to 1.68% for the top-9 accuracy. The top-10 accuracy stays constant, because the re-ranking cannot choose from outside the top-10 candidates. The MRR improves from 0.296 to 0.323 (9.15%).

Algorithm 2: Evaluation Procedure (per Data Split)

Input: a train- and test split of the evaluation corpus C_{train}, C_{test}, each including QA-pairs as tuples (q, a^+); the pre-trained baseline QA model for initial ranking r and the untrained re-ranking model r'.
Output: evaluation metrics.

```
begin
    // training of the re-ranking model
    X ⟵ [];
    y ⟵ [];
    foreach (q, a+) ∈ C_train do
        R ⟵ r.rank(q) ;                          // R contains top-10 results
        if a+ ∉ R) then
            continue with next QA pair
        else
            // add positive sample
            c+ ⟵ R[a+] ;                          // confidence for a+
            X.append(features_from(q, a+, c+));
            y.append(1);
            // add negative sample
            a- ⟵ random a ∈ R \ a+;
            c- ⟵ R[a-];
            X.append(features_from(q, a-, c-));
            y.append(0);
    r'.train(X, y);
    // evaluation of the re-ranking model
    results ⟵ ∅;
    foreach (q, a+) ∈ C_test do
        R ⟵ r.rank(q) ;                          // top-10 baseline ranking
        R' ⟵ r'.rank(q, R) ;                     // apply re-ranking
        results.append(R');
    return compute_metrics(results)
```

7 Discussion

Our results indicate that the accuracy of the described QA system benefits from our re-ranking approach. Hence, it can be applied to improve the performance of already deployed QA systems that provide a top-10 ranking with confidences as output. However, the performance gain is small, which might have several reasons. For example, we did not integrate spell-checking in our re-ranking method which proved to be effective in our baseline evaluation. Further, the re-ranking model is based on very simple features. It would be interesting to investigate the impact of more advanced features, or models, on the ranking performance (e.g., word embeddings [14] and deep neural networks for learning similarity functions [8, 15]). Nevertheless, as can be seen in examples 1, 2 and 4 in Table 1, high-ranked but incorrect answers are often meaningful with respect to the query: the setting in our evaluation is overcritical,

Table 3 Performance metrics of the baseline QA pipeline and our re-ranking method ($n = 3084$)

Metric	Method		Relative improvement (%)
	Baseline QA	*Re-ranking*	
Top-1 accuracy	0.208	0.239	14.83
Top-2 accuracy	0.299	0.334	11.84
Top-3 accuracy	0.353	0.384	8.99
Top-4 accuracy	0.391	0.415	6.31
Top-5 accuracy	0.417	0.44	5.59
Top-6 accuracy	0.438	0.459	4.83
Top-7 accuracy	0.454	0.471	3.74
Top-8 accuracy	0.466	0.48	3.02
Top-9 accuracy	0.478	0.486	1.68
Top-10 accuracy	0.494	0.494	0.00
MRR@10	0.296	0.323	9.15

because we count incorrect, but meaningful answers as negative result. A major limitation is that the re-ranking algorithm cannot choose answer candidates beyond the top-10 results. It would be interesting to classify whether an answer is present in the top-10 or not. If not, the algorithm could search outside the top-10 results. Such a meta-model can also be used to estimate weaknesses of the QA model: it can determine topics that regularly fail, for instance, to guide data labelling for a targeted improvement of the model, also known as active learning [22], and in combination with techniques from semi-supervised learning [6, 9].

Data labelling and incremental model improvement can be scaled by crowdsourcing. Examples include the parallel supervision of re-ranking results and targeted model improvement as human oracles in an active learning setting. Results from crowd-supervised re-ranking allows us to train improved re-ranking models [11, 19], but also a meta-model that detects queries which are prone to error. The logs of a deployed chatbot, that contain actual user queries, can be efficiently analysed using such a meta-model to guide the sample selection for costly human data augmentation and creation. An example of a crowdsourcing approach that could be applied to our QA system and data, with search logs can be found in [2].

8 Conclusion

We implemented a simple re-ranking method and showed that it can effectively improve the performance of QA systems after deployment. Our approach includes the top-10 answer candidates and confidences of the initial ranking for selecting better answers. Promising directions for future work include the investigation of more advanced ranking approaches for increasing the performance gain and continuous improvements through crowdsourcing and active learning.

Acknowledgements This work is supported by the German Federal Ministry of Education and Research (BMBF) in the Software Campus project SciBot and by EIT Digital (H2020) in the ERICS project.

References

1. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X (2015) TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software available from tensorflow.org
2. Bernstein MS, Teevan J, Dumais S, Liebling D, Horvitz E (2012) Direct answers for search queries in the long tail. In: Proceedings of the 2012 ACM annual conference on human factors in computing systems - CHI '12, pp 237, New York, USA. ACM Press
3. Bird S, Klein E, Loper E (2009) Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media Inc, Sebastopol
4. Bocklisch T, Faulkner J, Pawlowski N, Nichol A (2017) Rasa: open source language understanding and dialogue management. 12
5. Buck C, Bulian J, Ciaramita M, Gesmundo A, Houlsby N, Gajewski W, Wang W (2017) Ask the right questions: active question reformulation with reinforcement learning. arXiv:1705.0
6. Chang JC, Kittur A, Hahn N (2016) Alloy: clustering with crowds and computation. In: Proceedings of the 2016 CHI conference on human factors in computing systems - CHI '16, pp 3180–3191, New York, USA. ACM Press
7. Chen D, Fisch A, Weston J, Bordes A (2017) Reading wikipedia to answer open-domain questions. In: Proceedings of the 55th annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 1870–1879. ACL
8. Das A, Yenala H, Chinnakotla M, Shrivastava M (2016) Together we stand: siamese networks for similar question retrieval. In: Proceedings of the 54th annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 378–387, Stroudsburg, PA, 2016. Association for Computational Linguistics
9. Dhingra B, Danish D, Rajagopal D (2018) Simple and effective semi-supervised question answering. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, Volume 2 (Short Papers), pp 582–587. ACL
10. Eric M, Manning CD (2017) Key-value retrieval networks for task-oriented dialogue. In: Proceedings of the 18th annual SIGdial meeting on discourse and dialogue, pp 37–49, Saarbrücken, Germany. Association for Computational Linguistics
11. 'Kenneth' Huang T-H, Chang JC, Bigham JP (2018) Evorus: a crowd-powered conversational assistant built to automate itself over time. 1
12. Kenneth Huang T-H, Lasecki WS, Bigham JP (2015) Guardian: a crowd-powered spoken dialog system for web APIs. In: Third AAAI conference on human computation and crowdsourcing
13. Joshi M, Choi E, Weld DS, Zettlemoyer L (2017) TriviaQA: a large scale distantly supervised challenge dataset for reading comprehension. In: Barzilay R, Kan M-Y (eds) Proceedings of the 55th annual meeting of the association for computational linguistics, ACL 2017, Volume 1: Long Papers, pp 1601–1611. ACL
14. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. 1

15. Minaee S, Liu Z (2017) Automatic question-answering using a deep similarity neural network. 8
16. Oraby S, Gundecha P, Mahmud J, Bhuiyan M, Akkiraju R (2017) "How May I Help You?": modeling twitter customer service conversations using fine-grained dialogue acts. In: Proceedings of the 22nd international conference on intelligent user interfaces - IUI '17, pp 343–355, New York, USA, 2017. ACM Press
17. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learning Res 12:2825–2830
18. Rajpurkar P, Zhang J, Lopyrev K, Liang P (2016) Squad: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp 2383–2392. ACL
19. Savenkov D, Agichtein E (2016) CRQA: crowd-powered real-time automatic question answering system. In: Fourth AAAI conference on human computation and crowdsourcing
20. Serban IV, Sordoni A, Bengio Y, Courville A, Pineau J (2015) Building end-to-end dialogue systems using generative Hierarchical neural network models. 7
21. Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2015) A survey of available corpora for building data-driven dialogue systems. 12
22. Burr S (2010) Active learning literature survey. Univ Wisconsin Madison 52(55–66):11
23. Sonntag D (2008) On introspection, metacognitive control and augmented data mining live cycles
24. Sonntag D (2009) Introspection and adaptable model integration for dialogue-based question answering. In: IJCAI, pp 1549–1554
25. Sonntag D (2010) Ontologies and adaptivity in dialogue for question answering. Studies on the semantic web, 1st edn, vol 4. AKA and IOS Press, Heidelberg
26. Sonntag D, Engel R, Herzog G, Pfalzgraf A, Pfleger N, Romanelli M, Reithinger N (2007) Smartweb handheld - multimodal interaction with ontological knowledge bases and semantic web services. In: Artifical intelligence for human computing, ICMI 2006 and IJCAI 2007 international workshops, Banff, Canada, November 3, 2006, Hyderabad, India, January 6, 2007, Revised selected and invited papers, pp 272–295
27. Williams JD, Asadi K, Zweig G (2017) Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In: Proceedings of the 55th annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 665–677, Stroudsburg, PA, USA. Association for Computational Linguistics
28. Wu L, Fisch A, Chopra S, Adams K, Bordes A, Weston J (2017) Starspace: embed all the things! arXiv:1709.03856
29. Xue X, Jeon J, Croft WB (2008) Retrieval models for question and answer archives. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval - SIGIR '08, pp 475, New York, USA, 2008. ACM Press

Measuring Catastrophic Forgetting in Visual Question Answering

Claudio Greco, Barbara Plank, Raquel Fernández, and Raffaella Bernardi

Abstract Catastrophic forgetting is a ubiquitous problem for the current generation of Artificial Neural Networks: When a network is asked to learn multiple tasks in a sequence, it fails dramatically as it tends to forget past knowledge. Little is known on how far multimodal conversational agents suffer from this phenomenon. In this paper, we study the problem of catastrophic forgetting in Visual Question Answering (VQA) and propose experiments in which we analyze pairs of tasks based on CLEVR, a dataset requiring different skills which involve visual or linguistic knowledge. Our results show that dramatic forgetting is at place in VQA, calling for studies on how multimodal models can be enhanced with continual learning methods.

1 Introduction

Artificial Neural Networks (ANNs) have brought steep advances to fields such as speech technology, computer vision, and natural language processing, and lately also to visually grounded dialogue systems (e.g., [2]). However, there exists a fundamental open challenge for the development of truly intelligent artificial intelligence: Each

C. Greco (✉)
CIMeC, University of Trento, Rovereto, Italy
e-mail: claudio.greco@unitn.it

B. Plank
Department of Computer Science, IT University of Copenhagen, Copenhagen, Denmark
e-mail: bplank@itu.dk

R. Fernández
ILLC, University of Amsterdam, Amsterdam, The Netherlands
e-mail: raquel.fernandez@uva.nl

R. Bernardi
CIMeC, University of Trento, Rovereto, Italy
e-mail: raffaella.bernardi@unitn.it

DISI, University of Trento, Povo, Italy

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_35

time a task has to be learned, the model has to undergo a new expensive training phase using a huge dataset. This is in stark contrast to humans, who do not learn from scratch for each new task, but instead they build on previous experience, they incrementally refine their skills during their lifetime, and they typically learn from easier contexts first. The capability of machine learning models to continuously learn over time by accommodating new information while retaining their past memories is referred to as *continual learning* [11]. Most models, when trained sequentially on new tasks, forget how to perform the previously learned tasks. This phenomenon, called *catastrophic forgetting*, is prominent in ANNs [8].

Little is known on how much visually grounded conversational agents—a long standing challenge of artificial intelligence—suffer from this phenomenon. The task requires being able to master different skills both at the language and visual reasoning levels and at their interface. In the long run, we want to develop multimodal conversational agents which are able to learn over time without forgetting. As a first step, in this paper we study catastrophic forgetting for Visual Question Answering (VQA), which requires to answer natural language questions about images.

2 Related Work

Continual learning has been mostly studied in computer vision. Popular benchmarks for continual learning methods are tasks involving digit classification, including Split MNIST [14], or object classification [14], where object classes are learned incrementally. A more complex task has been studied in [6], where the model learns to play ATARI games incrementally. As far as we know, catastrophic forgetting in the domain of visually grounded dialogue has not been studied yet. An approach studying forgetting in the context of VQA is [10], where a model able to perform strong generalization has been shown to forget after having been fine-tuned on data including images where objects have different colors than those previously seen. We take this work as our starting point and extend it to consider different types of questions besides different colors, as explained in the next section.

In transfer learning, an interesting question is to uncover relationships between tasks, a problem that has been studied recently by [13] among visual tasks. We have not explored this issue yet, but we paid attention to the relationship between tasks in our experimental design by aiming to compare tasks that differ with respect to their visual versus linguistic input (Experiment 1 vs. Experiment 2 below). In the study of continual learning for multimodal models, an important question is which modality they forget the most. Hence, we focus on task pairs which differ either on visual or linguistic reasoning skills. Within the research community on dialogue, it is well-known that children have more difficulty learning polar (yes/no) questions than wh-questions [9]. Our comparison of linguistic skills in Experiment 2 is motivated by this finding.

Continual learning methods for classification tasks can be tested in two different settings: single- and multi-head. In the former, the output space consists of all the known labels, whereas in the latter the output space consists of only the labels of the current task and the model predictions must belong to the classes of that task. At test-time, in single-head the task identifier is unknown, whereas in multi-head it is given. Most of the current continual methods are evaluated in a multi-head setting, in which they perform much better than in single-head [1]. However, single-head is much more realistic, as models have to learn how to discriminate between tasks without any task identifier at test-time. Hence, we will adopt a single-head setting.

A comprehensive set of evaluation measures taking into account forgetting, transfer, and computational efficiency has been proposed by [3]. Furthermore, [1] proposes intransigence, which measures the inability to learn new tasks. The problem of intransigence is much more prominent in the single-head setting because the model must learn how to distinguish the current task from the previously learned ones.

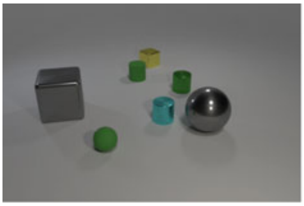
3 Datasets, Experiments, and Evaluation Metrics

To measure the effect of catastrophic forgetting in VQA we carry out an analysis on pairs of tasks. We study whether multimodal models forget previously learned tasks when the task pairs differ with respect to the visual or the linguistic data involved.

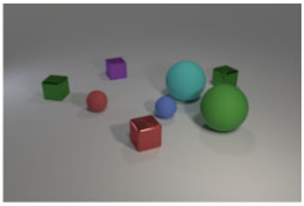
Datasets We take CLEVR [4] as starting point since it is a diagnostic dataset to assess model acquisition of visual reasoning skills. CLEVR is a VQA dataset which includes synthetic images and automatically generated natural language questions about them. Each image is associated with a scene graph specifying objects and attributes, whereas each question is derived by a Functional Program (FP) using the scene graph of the respective image to obtain the correct answer to the question. FPs are composed of simple functions corresponding to elementary operations (skills) in visual reasoning, such as querying object attributes or comparing values (see details below). Questions are categorized by their type, which is defined by the outermost function in their FP. In CLEVR there are two sizes, eight colors, two materials, and three shapes. For instance, the question shown in Fig. 1 (left) is a question of type "query_attribute". The FP associated to the question includes different operators involving various skills, such as 'query_material', 'filter_size' with parameter 'large', 'same_shape', or 'filter_color' with parameter 'yellow'.

Experiments We perform two experiments, which are described as follows:

EXPERIMENT 1: ITEMS SWAP COLORS In order to investigate the role of visual data in catastrophic forgetting, we use CLEVR-CoGenT released by [4] to evaluate how models learn compositional concepts that generalize. The model has to learn to answer all question types in CLEVR, but in the data used in Task A all cubes are gray, blue, brown, or yellow and all cylinders are red, green, purple, or cyan; while in the data used in Task B, cubes and cylinders swap color palettes. Both datasets contain spheres of all colors. Figure 1 shows examples of images of the two tasks.



Q: What is the material of the large object that is the same shape as the tiny yellow thing?
A: Metal



Q: Does the cyan ball have the same material as the tiny red thing that is behind the blue sphere?
A: Yes

Fig. 1 In EXP.1, T_A does not contain images like the one on the right (and viceversa for T_B). In EXP.2, T_A does not contain yes/no-questions like the one on the right (and viceversa for T_B)

EXPERIMENT 2: QUESTIONS CHANGE TYPES In order to investigate the role of different linguistic input, we created two tasks with wh- vs. yes/no questions about attributes. Task A contains only "query_attribute" (viz., wh) questions and Task B contains only "equality" (viz., yes/no) question types as illustrated in Fig. 1.

We build a training, a validation, and a test set for each of the experiments and each of the tasks. Since the original CLEVR test set does not contain ground-truth answers, we split the original validation set into validation and test set, accordingly. For Task B of EXPERIMENT 1, we also build a training set from the original validation set, because CLEVR-CoGenT does not have a training set for Task B.

For EXPERIMENT 1 the final train/validation/test split contains resp. 699,960, 75,000 and 75,000 instances for Task A and resp. 49,997, 49,997, and 49,997 for Task B. For EXPERIMENT 2, the same splits contain resp. 251,749, 26,960, and 26,774 instances for Task A and 141,596, 15,121, and 15,424 instances for Task B. We have analyzed the frequency of the functions in the FPs of the questions: in both experiments there is a huge overlap between the functions of the two tasks.

Evaluation We consider the following evaluation measures: *Accuracy* (*Acc*) and *Remembering* (*Rem*) [3] and *Intransigence* [1] (*Int*). *Acc* measures the average accuracy of the models on the learned tasks, *Rem* measures how much the model remembers how to perform previous tasks. *Int* measures how much the model is unable to learn new tasks. We also compute an overall score (*Overall*) defined as the average of *Acc*, *Rem*, and the difference between 1 and the value of *Int* normalized to lie in the range [0, 1]. If a model is heavily regularized towards preserving past knowledge, it will have high *Intransigence* as it will forget less but it will also be less capable of learning new tasks. Higher values for *Accuracy* and *Remembering* are better, whereas higher values for *Intransigence* are worse.

4 Models, Settings, and Results

Models We take the LSTM + CNN + SA architecture shown in [4] as starting point. It encodes questions with Long Short-Term Memories (LSTMs) and images with a ResNet-101 Convolutional Neural Network (CNN) pretrained on ImageNet. The visual and linguistics representations are combined using Spatial Attention (SA) to focus on the most salient objects and properties as in [12]. A Multilayer Perceptron (MLP) receiving as input the resulting features combined using SA predicts the answer distribution. In order to measure the amount of catastrophic forgetting in the two experiments, we consider the *Naive* and *Cumulative* baselines of [7]. For *Naive*, the model is fine-tuned across tasks: it is first trained on Task A and then on Task B starting from the previously learned parameters. For *Cumulative*, the model is first trained on Task A and then cumulatively on examples from Task A and Task B.

Settings The models were trained using Adam [5] with a learning rate of 0.0005 with early stopping. Word embeddings had a size of 300. RNNs had two hidden layers and LSTM cells had a size of 1024. MLPs had one hidden layer of size 1024. We use a single-head setup in which the output space consists of all the answers in CLEVR. As mentioned earlier, this is a more realistic setup, as humans would automatically understand which skills to exploit according to the kind of question.

Results Table 1 reports the results of the two experiments. First of all, we notice a difference in the difficulty between the task pairs: In EXPERIMENT 2 Task B (Yes/No-questions) is much harder than Task A (0.521 vs. 0.834, respectively), while this is less pronounced in EXPERIMENT 1 (0.597 vs. 0.797).[1]

Regarding differences and similarities between the two baseline models, the *Cumulative* model by nature cannot forget, since it always sees data from Task A. Hence, its performance is high in both experiments when trained on AB and tested on A. In contrast, the *Naive* model is affected by a small amount of forgetting in EXPERIMENT 1 (from 0.797 to 0.775) and a dramatic one in EXPERIMENT 2 (from 0.834 to 0.001). In EXPERIMENT 1, both models profit from sequential learning: they increase their performance on Task B if they have been trained on Task A before (*Naive*) or with alternate batches (*Cumulative*): from 0.597 to 0.764. In EXPERIMENT 2, training on Task A does not contribute to increase the performance on Task B for the *Naive* model (0.521 vs. 0.517), while the *Cumulative* model benefits substantially from being exposed to both tasks, reaching 0.749.

The measures on the right side of Table 1 provide an overview of the models' performance. Regarding *Accuracy*, in EXPERIMENT 1, the difference between *Naive* and *Cumulative* is small, indicating that both models perform well on the tasks they are trained on (0.785 vs. 0.779). This does not happen in EXPERIMENT 2, where *Naive* has much lower performance, mostly because its accuracy on Task A is close to zero after its training on Task B. Regarding *Remembering*, in EXPERIMENT 1, *Naive* has

[1] In EXPERIMENT 1, Task A and B are trained on different amounts of data (see Sect. 3). We have trained the model on Task A with the same amount of data as Task B obtaining very similar performance: 0.566.

Table 1 Results of the two experiments (EXP 1 & EXP 2). 'Train A Test A' ('Train B Test B'): accuracy when training and testing on Task A (B); 'Train AB Test A': accuracy when training on A and B and testing on A; 'Train AB Test B': accuracy when training on A and B and testing on B. The overall metrics on the right are computed according to 'Train A Test A', 'Train AB Test A' and 'Train AB Test B'

		Train A Test A	Train B Test B	Train AB Test A	Train AB Test B	*Acc*	*Rem*	*Int*	Overall
EXP1	Naive	0.797	0.597	0.775	0.764	0.779	0.978	0.000	0.752
	Cumul.			0.792	0.764	0.785	0.995	0.000	0.760
EXP2	Naive	0.834	0.521	0.001	0.517	0.451	0.167	0.233	0.334
	Cumul.			0.881	0.749	0.821	1.000	0.000	0.774

a lower score than *Cumulative*, as it partially forgets Task A after its training on Task B. The same happens in EXPERIMENT 2, where *Naive* has a much lower score than *Cumulative*. This suggests that in EXPERIMENT 2 the first task (wh-questions) is more easily forgotten by the *Naive* model. Regarding *Intransigence*, in EXPERIMENT 1 both models have zero scores, which indicates that they are able to learn the new task easily. This is not the case in EXPERIMENT 2, where *Naive* has a higher *Int* score than *Cumulative*, showing that *Naive* is not able to learn Task B as easily as *Cumulative* does. Thus, when the difference lays in the type of linguistic input as in EXPERIMENT 2, the model has more difficulty in learning new tasks. Finally, according to the Overall score, *Naive* is better than *Cumulative* in both experiments, but more so in EXPERIMENT 2.

5 Conclusion

In this paper, we assessed to what extent a multimodal model suffers from catastrophic forgetting in a VQA task. We built tasks involving different visual and linguistic characteristics and investigated whether multimodal models trained to solve some tasks remembered how to solve previously learned ones. Our results show that dramatic forgetting is at place in VQA, particularly when tasks involve different types of linguistic input, and call for studies on how visually grounded models can be enhanced with continual learning methods.

References

1. Chaudhry A, Dokania PK, Ajanthan T, Torr P (2018) Understanding forgetting and intransigence. In: ECCV, Riemannian walk for incremental learning
2. Das A, Kottur S, Gupta K, Singh A, Yadav D, Moura JMF, Parikh D, Batra D (2017) Visual dialog. In: CVPR
3. Díaz-Rodríguez N, Lomonaco V, Filliat D, Maltoni D (2018) Don't forget, there is more than forgetting: new metrics for continual learning. In: Workshop on continual learning, NeurIPS
4. Johnson J, Hariharan B, van der Maaten L, Fei-Fei L, Lawrence Zitnick C, Girshick R (2017) CLEVR: a diagnostic dataset for compositional language and elementary visual reasoning. In: CVPR
5. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
6. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A et al (2017) Overcoming catastrophic forgetting in neural networks. PNAS
7. Maltoni D, Lomonaco V (2018) Continuous learning in single-incremental-task scenarios. arXiv:1806.08568
8. McClelland JL, McNaughton BL, O'reilly RC (1995) Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. Psychol Rev 102(3)
9. Moradlou S, Ginzburg J (2014) Learning to understand questions. In: SemDial
10. Perez E, Strub F, De Vries H, Dumoulin V (2018) Visual reasoning with a general conditioning layer. In: AAAI, Film
11. Ring M (1997) CHILD: a first step towards continual learning. Mach Learning 28(1)
12. Yang Z, He X, Gao J, Deng L, Smola A (2016) Stacked attention networks for image question answering. In: CVPR
13. Zamir AR, Sax A, Shen W, Guibas L, Malik J, Savarese S (2018) Disentangling task transfer learning. In: CVPR, Taskonomy
14. Zenke F, Poole B, Ganguli S (2017) Continual learning through synaptic intelligence. In: ICML

Position Paper: Brain Signal-Based Dialogue Systems

Odette Scharenborg and Mark Hasegawa-Johnson

Abstract This position paper focuses on the problem of building dialogue systems for people who have lost the ability to communicate via speech, e.g., patients of locked-in syndrome or severely disabled people. In order for such people to communicate to other people and computers, dialogue systems that are based on brain responses to (imagined) speech are needed. A speech-based dialogue system typically consists of an automatic speech recognition module and a speech synthesis module. In order to build a dialogue system that is able to work on the basis of brain signals, a system needs to be developed that is able to recognize speech imagined by a person and can synthesize speech from imagined speech. This paper proposes combining new and emerging technology on neural speech recognition and auditory stimulus construction from brain signals to build brain signal-based dialogue systems. Such systems have a potentially large impact on society.

1 Introduction

A speech-based dialogue system typically takes in a spoken utterance by the user on the basis of which an action from the dialogue system follows. Communication from the dialogue system to the user occurs either in text or using synthesized speech. People who have lost the ability to communicate via speech or sign language (e.g., severely paralyzed people or patients of locked-in syndrome [1]) cannot use existing dialogue systems, nor are they able to communicate with other people. In order for them to communicate, brain-computer interfaces are needed [2]. These brain-computer interfaces should be able to convert the intended message from the neural activity in the brain areas related to speech processing and production into an action

O. Scharenborg (✉)
Multimedia Computing Group, Delft University of Technology, Delft, The Netherlands
e-mail: O.E.Scharenborg@tudelft.nl

M. Hasegawa-Johnson
University of Illinois and Beckman Institute, Urbana, Champaign, IL, USA
e-mail: jhasegaw@illinois.edu

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_36

carried out by the dialogue system or into text or synthesized speech in case of communication with another person [3].

Two approaches investigating the decoding of speech from neural signals can be distinguished: in the *overt* condition, listeners' neural signals when listening to speech are recorded and decoded; in the *covert* condition, listeners' neural signals are recorded while they imagine to speak and subsequently decoded. The former case is an important step to understand the relationship between the acoustic signal and the neural signal; the latter case is the situation that allows patients to communicate their thoughts and wishes and is the ultimate dream.

Neural signals: The most often used type of neural signal is electrocorticography (ECoG), which is an invasive methodology in which electrode arrays are placed directly onto the surface of the brain in patients. Electroencephalography (EEG) is less invasive as it 'only' requires wearing a cap with electrodes, making it a technique that is more user-friendly and cheaper than ECoG. A downside to using EEG signals compared to ECoG is that because EEG caps are placed on the outside of the skull, brain signals obtained with EEG are noisier and have a less good spatial resolution than ECoG signals. EEG signals however have good time resolution which is import in speech processing.

Overt condition: [4] presented a proof-of-concept neural speech recognition system, which used brain responses to continuous speech produced by two speakers obtained using ECoG from three patients receiving surgery related to epilepsy. Data from these three individuals were used to train three listener-dependent systems and a listener-independent system. The obtained phone error rates ranged from 70 to over 80% for the listener-dependent systems. A review of automatic speech recognition of different types of neural signals found that ECoG provided the best recognition results [5]. Although recognition is poor, these systems show that listener-independent linguistic information can be obtained from the ECoG signals.

Covert condition: The neural signals that give the best results in brain-computer interfaces are obtained using ECoG [6, 7, 8]. EEG signals have, however, with limited success been used to decode imagined articulation of two English vowels [9], three Dutch vowels [10], two Japanese vowels [11], and "yes" and "no" [12] with above chance accuracy. Although more research is needed before this technique can be fully used, for a dialogue system, "yes" and "no" are highly important words.

Auditory stimulus reconstruction: Auditory stimulus reconstruction is an inverse mapping technique which attempts to create an auditory signal from the neural signals [3, 13, 14, 15]. This technique can be used to convert the neural signals from a person listening to speech (overt condition) or the articulation of words imagined by a person (covert condition) into a temporal and spectral representation. Typically, in speech-based brain-computer interfaces, the neural activity to (imagined) speech is decoded into linguistic units such as phonemes or words or acoustic units such as the speech envelope or the magnitude spectrogram (see [14] for references), which can be synthesized as speech. Recently, [14] proposed to train a DNN to directly predict the parameters of the synthesizer from ECoG signals to covert speech rather than go via intermediate representations, so combining neural speech recognition and synthesis. This approach significantly outperformed a system which used an audiospectrogram as an intermediate unit.

2 Conclusion

The question whether it is possible to build dialogue systems for people who have lost the ability to communicate via speech using their brain responses to speech cannot yet be answered in the affirmative. However, initial building blocks are in place to build such systems, especially for the construction of dialogue systems which require "yes"/"no" answers. The ultimate goal is to make it possible for the patient to communicate his or her thoughts by imagining speech which subsequently can be synthesized, ideally including emotional and speaker-dependent characteristics. Because of user-friendliness, EEG-based technology is preferred over the invasive ECoG-based approach. More research is needed to improve the independent modules and integrate them into working dialogue systems.

References

1. Laureys S et al (2005) The locked-in syndrome: what is it like to be conscious but paralyzed and voiceless? Prog Brain Res 150:495–611
2. Sellers EW, Ryan DB, Hauser CK (2014) Noninvasive brain-computer interface enables communication after brainstem stroke. Sci Transl Med 6(257):257re7
3. Iljina O et al (2017) Neurolinguistic and machine-learning perspectives on direct speech BCIs for restoration of naturalistic communication. Brain-Comput Interfaces 4:186–199
4. Moses DA, Mesgarani N, Leonard MK, Chang EF (2016) Neural speech recognition: continuous phoneme decoding using spatiotemporal representations of human cortical activity. J Neural Eng 13:19
5. Herff C, Schultz T (2016) Automatic speech recognition from neural signals: a focused review. Front Neurosci 10:429
6. Martin S et al (2016) Word pair classification during imagined speech using direct brain recordings. Sci Rep 6:25803
7. Leuthardt EC et al (2011) Using the electrocorticographic speech network to control a brain–computer interface in humans. J Neural Eng 8
8. Pei X, Barbour DL, Leuthardt EC, Schalk G (2011) Decoding vowels and consonants in spoken and imagined words using electrocorticographic signals in humans. J Neural Eng 8:46028
9. Dasalla CS, Kambara H, Sato M, Koike Y (2009) Single-trial classification of vowel speech imagery using common spatial patterns. Neural Netw 22:1334–1339. https://doi.org/10.1016/j.neunet.2009.05.008
10. Hausfeld L, De Martino F, Bonte M, Formisano E (2012) Pattern analysis of EEG responses to speech and voice: influence of feature grouping. Neuroimage 59:3641–3651. https://doi.org/10.1016/j.neuroimage.2011.11.056
11. Natsue Y, Atsushi N, Nasreddine BA, Duk S, Hiroyuki K, Takashi H, Yasuharu K (2016) Decoding of covert vowel articulation using electroencephalography cortical currents. Front Neurosci 10:175. https://doi.org/10.3389/fnins.2016.00175

12. Lopez-Gordo MA, Fernandez E, Romero S, Pelayo F, Prieto A (2012) An auditory brain-computer interface evoked by natural speech. J Neural Eng 9:036013. https://doi.org/10.1088/1741-2560/9/3/036013
13. Pasley BN, et al (2012) Reconstructing speech from human auditory cortex. Plos Biol 10
14. Akbari AH, Khalighinejad B, Herrero JL, Mehta AD, Mesgarani N (2018) Towards reconstructing intelligible speech from the human auditory cortex. Sci Rep 9(1):874. https://doi.org/10.1038/s41598-018-37359-z
15. Chakrabarti S, Sandberg HM, Brumberg JS, Krusienski DJ (2015) Progress in speech decoding from the electrocorticogram. Biomed Eng Lett 5:10–21

First Leap Towards Development of Dialogue System for Autonomous Bus

Maulik C. Madhavi, Tong Zhan, Haizhou Li, and Min Yuan

Abstract This paper describes the dialogue system for the autonomous bus. Without driver onboard in an autonomous bus, a passenger needs someone to talk to when in need. In that scenario, the dialogue system in this work helps a passenger to manage travel plan. The system is designed to work in both chat-oriented and goal-oriented conversations. The internal design is rule-based utterance matching. We also describe the database, which can be easily expandable by the administrator for future development. The dialogue system deployment on android smartphone interface is demonstrated in this paper.

1 Introduction

Recent technology related to self-driving cars and autonomous vehicle offers several advantages over conventional transportation such as less transportation cost and 24 h facility [5]. In a conventional bus transportation service, onboard passengers usually move to the front to speak to the bus driver (captain) when there is a need. The autonomous bus (AB) would need to have additional communication/interaction channels for the passengers in an absence of driver respond to the query. This communication can be helpful for multiple purposes, for example, to inform passengers the status and navigation plan of autonomous driving, to greet passengers, to alert

M. C. Madhavi (✉) · T. Zhan · H. Li · M. Yuan
Department of Electrical and Computer Engineering, National University of Singapore (NUS), Singapore, Singapore
e-mail: maulik.madhavi@nus.edu.sg

T. Zhan
e-mail: elezha@nus.edu.sg

H. Li
e-mail: haizhou.li@nus.edu.sg

M. Yuan
e-mail: eleyumi@nus.edu.sg

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_37

passengers of danger, and to respond to an emergency request. Moreover, people are getting more and more used to using conversational systems or chatbots to manage the schedule and access the information via speech and/or text input [3]. Thus, AB needs to have a communication system between the AB computer system (in particular dialogue system) and the onboard passengers.

The dialogue system is a natural, flexible and intuitive way to establish communication between user and machine or robot and has gained widespread applications in several places such as voice operated a call center, entertainment, banking applications, health-care etc. [4]. Similarly, in AB, dialogue system can be helpful to serve as a role of human driver. For the deployment, we have two possibilities, namely, onboard kiosk and smartphone application. The onboard kiosk can be at a fixed position on a bus to provide dialogue service. The main limitation of the onboard kiosk is its inability to serve multiple passengers at a time. The smartphone app allows multiple passengers to chat with the AB computer (or dialogue system) concurrently. Furthermore, passenger can speak or type to their own smartphones at the comfort of their seat. The typing facility is also very useful for people having speech or hearing related disabilities. This work is partly motivated by the project, CMU Let's Go, where the dialogue system is developed over telephone communication [6]. However, it is different from Let's Go because smartphone-based voice interface plays a major role in the dialogue execution. In the next section, we will discuss the framework of the spoken dialogue system.

2 Dialogue System Framework

From the implementation viewpoint, similar to conventional spoken dialogue system, the proposed bus information agent can handle two types of conversations, namely, question-answer (Q&A) based and goal oriented [1]. The Q&A based conversation is the context-free conversation between the passenger and the virtual bus agent. The goal-oriented conversation uses the past queries to understand the passenger's goal and help him/her to accomplish the goal. The Q&A based conversations are mainly for general conversation for greeting, acknowledgment, bus management rule-sharing, etc. The goal-driven conversations are related to bus stop related queries, which can be addressed by use of AB data.

Figure 1 show subsystems in a block diagram of spoken dialogue system framework. The dialogue system contains the three subsystems, namely, Question and answer (Q&A) subsystem, multiple goal driven subsystem and Google knowledge interface. The dialogue classification selects the appropriate subsystem with an intent recognition module. The Q&A subsystem maintains the general queries with Q&A database. If the query is in the functional format, the response to be generated depends on the hidden entity and intent of the query, the conversation is of goal-driven type. If the query is related to bus station related, then the goal-driven system takes care the query with the help of AB database. We used past three dialogue intents to store the dialogue history, which is important in goal-driven system. If the query is more

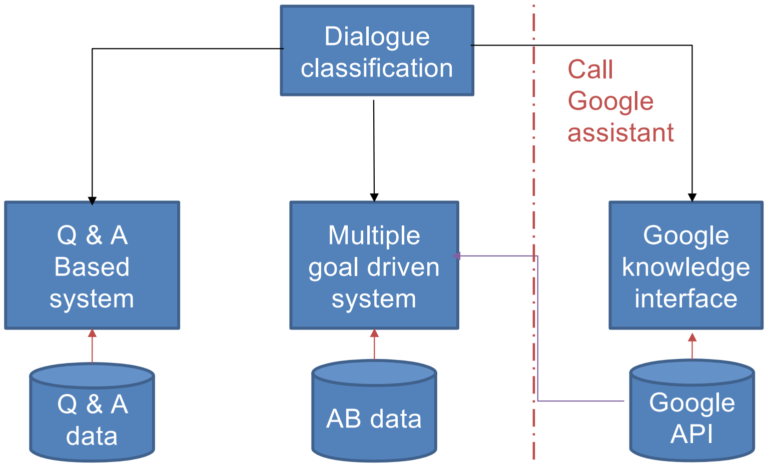


Fig. 1 Three sub-systems of AB spoken dialogue system, namely, Q&A subsystem, goal-driven subsystem, and Google linkage subsystem

general and not answerable by AB database, Google knowledge interface handles such kind of queries, such as, 'nearby shopping mall', etc. In Fig. 1, the curtain separates the Google interface to other two systems as to indicate that knowledge base of Google interface is not managed by our dialogue system. Table 1 shows the examples of conversations between user and agent. In the goal-driven conversation, for payment query, the response requires two slots from users, namely station and card type, which are asked by an agent in during conversations. The database for Q&A and multi goal-driven dialogues is prepared with the help of a survey from 20 subjects.

In this work, the vector representation of query is matched against all the stored queries using cosine similarity metric and the candidate response is selected based on the maximum similarity. In order to represent the query utterance, the universal sentence encoding representation trained on large corpus with supervised learning is used [2]. We use a pre-trained infersent model to obtain 4096-dimensional vector representation for each query utterance. The sentence encoder used in this work was earlier tested for semantic textual similarity score [2].

2.1 Q&A Database

The spoken dialogue system of AB is supported by two types of databases associated with two types of conversations, namely, Q&A, and goal-driven. Q&A subsystem is supported by the list of question (query) and answer pairs. Table 2 shows the sample Q&A data used in our dialogue system. The scalability of Q&A database is important for system development and maintenance. We have proposed a Python

Table 1 Examples of two types of conversations, namely, Q&A type and goal-driven

Q&A type conversation	
User	Hello (Greeting)
Agent	Hello! Nice to meet you!
User	Who are you (Self introduction)
Agent	I am bus information agent
User	Can I stand on the upper decker? (Bus management rules)
Agent	For security reasons, you are not allowed to stand on upper decker
User	Thanks (Acknowledgment)
Agent	You are welcome
Goal-driven conversation (AB data)	
User	How do I Pay (Payment)
Agent	Where do you alight at?
User	Clementi station (Slot: destination)
Agent	Which card are you holding? Student, senior citizen or adult
User	Adult (Slot: card type)
Agent	Please pay 1.60 SGD
Goal-driven conversation (Google knowledge interface)	
User	Nearby shopping mall (Google custom search)
Agent	The Clementi Mall, The Star Vista, Rochester Mall, West Coast Plaza, Viva Vista Mall

Table 2 The sample set of Q&A data

Category	Query	Response
Greeting	Hello, Good morning	Hello! Nice to meet you
Acknowledge	Thank you, Thanks	You are welcome
Self-introduction	Please tell me your name	I am bus information agent
Rule-sta_name	Can you alert when to get off	For your information, the name of the station will appear on the screen
Rule-ticket_print	No tickets printed what should I do	Your feedback has been transferred to maintenance's team. Have a nice ride!
Rule-change	Do you have any change	Sorry, We do not provide change
Rule-upper_deck	Can we stand on the upper decker	For safety reasons, standing at the upper decker is prohibited. Please stand at the lower decker
Rule-trolley	I am with my trolley bag. Can I take onboard	Yes, you must fold up the trolley when needed

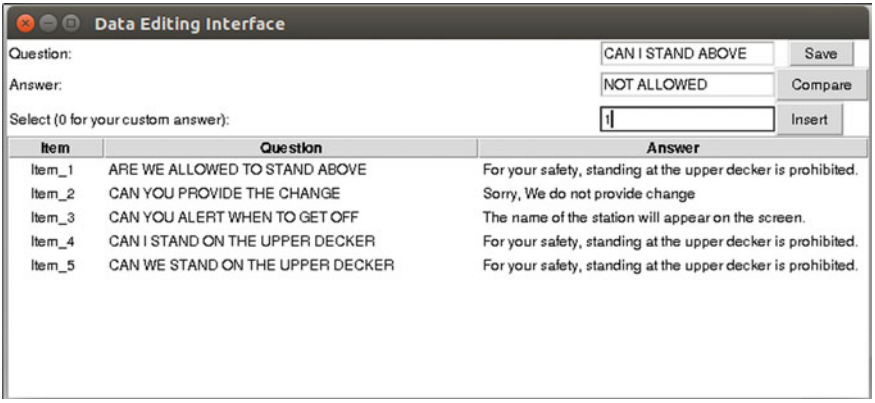


Fig. 2 The screenshot of Q&A edit interface for an administrator to scale up the database

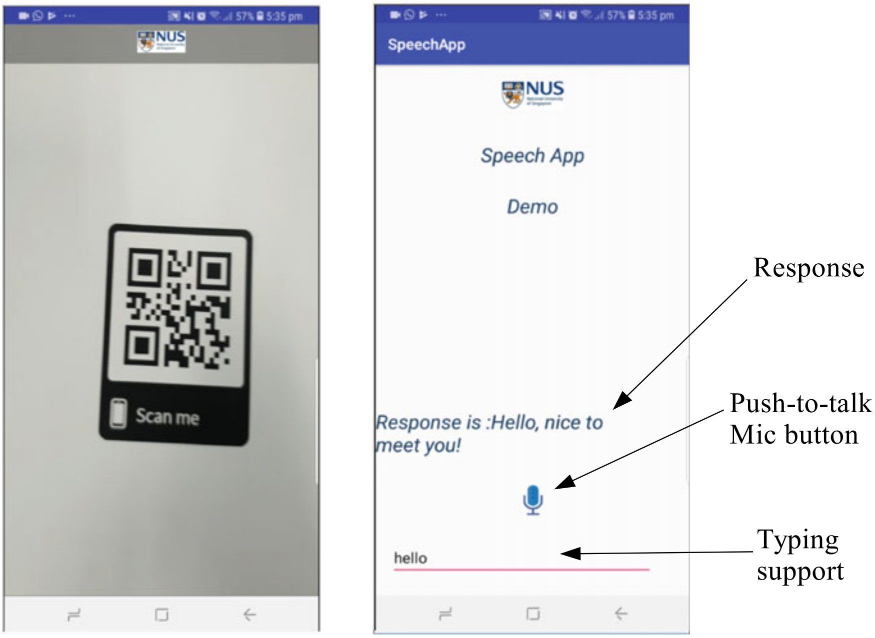
interface that allows administrators to augment the existing database with new Q&As. When receiving an input Q&A, the Python interface is able to automatically decide whether to merge with an existing entry or to create new entry. It also shows the similar five matching results to let the administrator select appropriate candidate. Thus, this interface provides ‘easy to edit and upgrade’ interface for non technical person. Figure 2 shows the screenshot of Q&A edit interface.

2.2 AB Database

This database is related to the autonomous bus service and useful while goal-drive conversation. AB data is linked to particular bus services. Each distinct QR code is responsible for loading the database of particular bus service. This database deals with the queries related to bus station information, fare related queries, etc. Thus, it manages goal-driven conversation as to achieve the goal of a user (passenger). The goal-driven conversations together with the dialogue history are maintained by hand-crafted rules. In addition to this data, the backend system is connected to the Google knowledge interface for custom search related to places.

3 Dialogue System Interface

For practical deployment, we have used socket communication for connecting the android smartphone to the dialogue system backend system. The dialogue supports both speech and text interface. The speech interface converts the user’s spoken input into text and we use in built Google speech recognition. The android smartphone



(a) QR code scan (b) Android UI

Fig. 3 Android smartphone interface to AB spoken dialogue system. Left panel shows the QR code scanning interface. Right panel shows the android UI supported by speech and text inputs

interface is shown in Fig. 3. The QR code scanning registers the user to communicate the dialogue system (refer Fig. 3a). The UI provides both speech and text interface for communicating the backend dialogue system (refer Fig. 3b).

4 Evaluation of Dialogue System

The performance of dialogue system depends on the correct identification of intent and entity. The evaluation is done at two levels, namely, dialogue classification-level and subsystem-level. At dialogue classification-level, we evaluate the classification performance as to how a test query is classified into one of the three subsystems. At subsystem-level, we evaluate the recognition performance within three subsystem assuming that the dialogue classification is correct. The cosine similarity between the vector representation of query from test set and train is performed to obtain the intent and entity. We have total 32 types of queries, which are having distinct intent and entity. The test query is said to be correctly classified if the response of the test query is the same as a matched query. In this experiment, we randomly selected 34 queries (20% of the total number of queries) as test queries from the total of 224

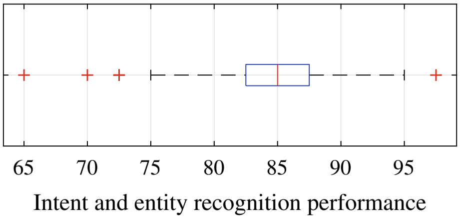


Fig. 4 Performance of intent and entity recognition in box-whisker plot (median is at 85.0%)

Table 3 Performance of intent and entity recognition

Level of evaluation	Subsystem	Performance (%)
Classification level	–	97.97
Subsystem level	Q&A	91.82
	Multi-goal	77.96
	Google knowledge	97.97
Overall	–	84.91

queries. We kept the remaining queries as training queries. For the experiments to be statistical significant, we conducted 1000 simulations and shown in Fig. 4. The performance of intent recognition is given in Table 3.

5 Summary and Conclusions

This paper presents the description of a spoken dialogue system for AB developed on android smartphone. The dialogue system supports two types of conversations, namely, Q&A and goal-driven conversations. We presented an overall design of the dialogue system with a data editing interface scheme for administration purpose. We evaluated our dialogue system in terms of joint intent and entity recognition performance. This work only scratches the surface of a possible design of the spoken dialogue system for AB. However, there are future scopes to involve machine learning based approaches for natural language understanding and dialogue policy. Further, we would like to evaluate the performance of the entire dialogue system.

Acknowledgements This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its LTA Urban Mobility Grand Challenge Program, Project Code UM01/002, ST Kinetics Autonomous Bus Trial.

References

1. Banchs RE, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In: Proceedings of the ACL 2012 system demonstrations. Association for Computational Linguistics, pp 37–42
2. Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A (2017) Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of the 2017 conference on empirical methods in natural language processing, EMNLP 2017, Copenhagen, Denmark, pp 670–680
3. Khan R, Das A (2017) Build better chatbots: a complete guide to getting started with chatbots. Apress
4. McTear MF (2002) Spoken dialogue technology: enabling the conversational user interface. ACM Comput Surv (CSUR) 34(1):90–169
5. Nielsen TAS, Haustein S (2018) On sceptics and enthusiasts: What are the expectations towards self-driving cars? Transport Policy 66:49–55
6. Raux A, Langner B, Bohus D, Black AW, Eskenazi M (2005) Let's go public! taking a spoken dialog system to the real world. In: Ninth European conference on speech communication and technology, Sept. 4–8, Lisbon, Portugal, pp 885–888

Dialogue Breakdown Detection

Overview of the Dialogue Breakdown Detection Challenge 4

Ryuichiro Higashinaka, Luis F. D'Haro, Bayan Abu Shawar, Rafael E. Banchs, Kotaro Funakoshi, Michimasa Inaba, Yuiko Tsunomori, Tetsuro Takahashi, and João Sedoc

Abstract To promote the research and development of dialogue breakdown detection for dialogue systems, we have been organizing a series of dialogue breakdown detection challenges to detect a system's inappropriate utterances that lead to dialogue breakdowns in chat-oriented dialogue. In this paper, we overview Dialogue Breakdown Detection Challenge 4 (DBDC4). As in the previous challenges, we used datasets in English and Japanese. Four teams participated in the challenge, in which all four teams worked on English, and two of the four teams worked on Japanese as well. This paper describes the task setting, evaluation metrics, and datasets for the challenge and the results of the submitted runs of the participants.

R. Higashinaka (✉)
NTT Media Intelligence Laboratories, NTT Corporation, Tokyo, Japan
e-mail: ryuichiro.higashinaka.tp@hco.ntt.co.jp

L. F. D'Haro
ETSI de Telecomunicación, Universidad Politécnica de Madrid, Madrid, Spain
e-mail: luisfernando.dharo@upm.es

B. Abu Shawar
College of Engineering, Al Ain University, Al Ain, United Arab Emirates
e-mail: bayan.abushawar@aau.ac.ae

R. E. Banchs
School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore
e-mail: rafael.banchs@gmail.com

K. Funakoshi
Laboratory for Future Interdisciplinary Research of Science and Technology, Tokyo Institute of Technology, Yokohama, Japan
e-mail: funakoshi@lr.pi.titech.ac.jp

M. Inaba
Artificial Intelligence eXploration Research Center, The University of Electro-Communications, Chofu, Japan
e-mail: m-inaba@uec.ac.jp

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_38

1 Introduction

Detecting dialogue breakdown [7] is the task of detecting whether a system utterance causes a dialogue breakdown (when users cannot continue a conversation with the system [5, 6, 9]) in a given dialogue context. The participants develop a dialogue breakdown detector that outputs a dialogue breakdown label (B: breakdown, PB: possible breakdown, or NB: not a breakdown) and a distribution of these labels. The labels are defined as follows.

(NB) Not a breakdown: It is easy to continue the conversation.
(PB) Possible breakdown: It is difficult to continue the conversation smoothly.
(B) Breakdown: It is difficult to continue the conversation.

In Dialogue Breakdown Detection Challenge 4 (DBDC4), as in Dialogue Breakdown Detection Challenge 3 (DBDC3), we prepared datasets in two languages: English and Japanese. Out of ten teams registered, four teams participated in this challenge. All four teams worked on English, and two of the four teams worked also on Japanese. In this paper, we describe the task setting, evaluation metrics, and datasets, and we give the results of the runs submitted by the participants.

2 Task Setting

Dialogue breakdown detection is detecting whether a system's utterances cause dialogue to break down in a given dialogue context [7]. For example, consider the following dialogue.

S1: It's nice to go shopping alone.
U1: I agree. It's nice.
S2: Shopping takes time.
U2: Window shopping is also fun.
S3: It's fun to go shopping with someone else.

Here, system utterances are prefixed by S and user utterances by U. The dialogue context is from S1 to U2, and the target utterance for dialogue breakdown detection is S3 (underlined). In this example, S3 will *likely* cause a dialogue breakdown because

Y. Tsunomori
Service Innovation Department, NTT DOCOMO INC., Tokyo, Japan
e-mail: yuiko.tsunomori.fc@nttdocomo.com

T. Takahashi
AI Deployment Project, Fujitsu Laboratories, LTD., Kawasaki, Japan
e-mail: takahashi.tet@jp.fujitsu.com

J. Sedoc
Department of Technology, Operations, and Statistics, New York University, New York, USA
e-mail: jsedoc@nyu.edu

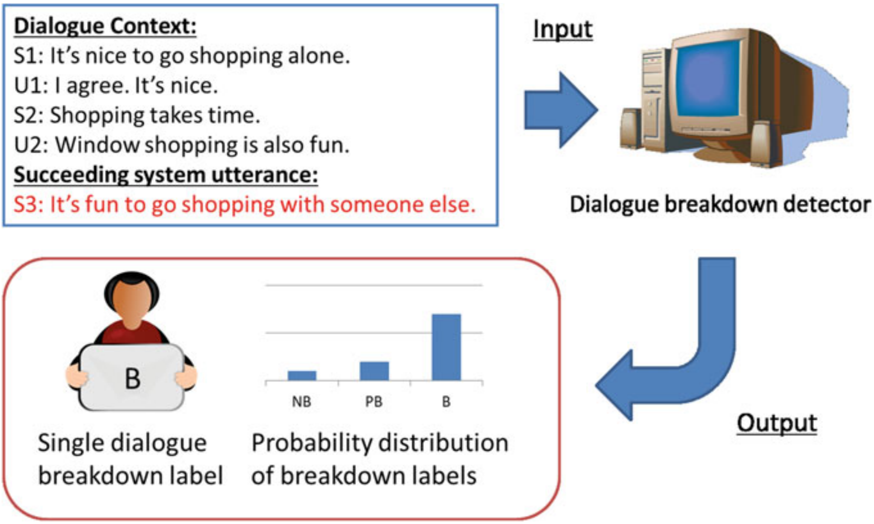


Fig. 1 Illustration of the task setting

S3 contradicts S1. Therefore, a detector that classifies this as a dialogue breakdown will be regarded as accurate. We say "likely" because a human's decision regarding dialogue breakdown is highly subjective, and deciding on a single gold label is difficult. Therefore, we use several annotators for dialogue breakdown annotation and refer to majority voting and their probability distribution.

The participants were given pairs of dialogue context and a succeeding system utterance; then, they submitted (1) a single dialogue breakdown label and (2) the probability distribution of breakdown labels for each pair (See Fig. 1). Although some utterances may exist after the target utterance, they cannot be used for prediction because we focus on avoiding dialogue breakdown rather than on recovery. Each participant could submit up to five "runs" for each language so several parameters for dialogue breakdown detection could be tested. Compared to the previous challenges, we increased the number of runs from three to five, reflecting the possible large parameter space when applying deep learning methods.

3 Evaluation Metrics

As in the previous challenges, we used classification-related and distribution-related evaluation metrics. In DBDC4, we emphasized mean squared error (MSE) related metrics, namely MSE(NB, PB, B) and MSE(NB+PB, B), which we found were better than other metrics; we enumerated possible evaluation metrics and ranked them based on system ranking stability and discriminative power, which are the two criteria commonly used in information retrieval research to derive evaluation metrics. See [15] for more details.

3.1 Classification-Related Metrics

We evaluated the accuracy in classifying dialogue breakdown labels with classification-related metrics. We calculated the accuracy by comparing the output of the detector and the gold label determined by majority voting. We used a threshold t to obtain the gold label by first finding the majority label and checking if the ratio of that label is above t; if so, the gold label becomes that label and NB otherwise. We used the following metrics.

- Accuracy: the number of correctly classified labels divided by the total number of labels to be classified.
- Precision, Recall, F-measure (B): the precision, recall, and F-measure for the classification of B labels.
- Precision, Recall, F-measure (PB+B): The precision, recall, and F-measure for the classification of PB + B labels; that is, PB and B labels are treated as a single label.

These metrics can provide intuitive results for the detection of dialogue breakdowns because they are used to directly evaluate whether dialogue breakdowns are correctly classified or not. However, the choice of an appropriate t value remains an issue. In this challenge, we used $t = 0.0$, which means simple majority voting.

3.2 Distribution-Related Metrics

We used distribution-related metrics to evaluate the similarity between the distributions of dialogue breakdown labels, calculated by comparing the predicted distribution of the labels with that of the gold labels. We calculated these values for each utterance and used the mean values for evaluation. We used the following metrics.

- JS divergence (NB, PB, B): distance between the predicted distribution of the three labels and that of the gold labels calculated by the Jensen–Shannon divergence.
- JS divergence (NB, PB+B): JS divergence when PB and B are regarded as a single label.
- JS divergence (NB+PB, B): JS divergence when NB and PB are regarded as a single label.
- Mean squared error (NB, PB, B): distance between the predicted distribution of the three labels and that of the gold labels calculated by the mean squared error.
- Mean squared error (NB, PB+B): mean squared error when PB and B are regarded as a single label.

- Mean squared error (NB+PB, B): mean squared error when NB and PB are regarded as a single label.

These metrics are used to compare the distributions of the labels, enabling a direct comparison with the gold labels. These metrics are more reliable compared to the classification-related ones though not be as interpretable.

4 Datasets

We prepared datasets in English and Japanese. All the dialogues in the datasets are composed of alternately conversed human-chatbot text dialogue. All the system responses in the datasets were annotated with three dialogue breakdown labels: NB, PB, and B. We give the details of the datasets in each language below.

4.1 Datasets for English

The development and evaluation data consisted of dialogue sessions from two resources, the dialogue sessions made with the IRIS [1, 3] dialogue system and Conversational Intelligence Challenge 2 (ConvAI2) dataset.[1] Among the multiple datasets provided by ConvAI2, the dialogues provided as "data_tolokers.json" were used for DBDC4. "data_tolokers.json" has dialogue sessions made with multiple dialogue systems indicated by anonymous names such as "Bot 001." We selected dialogue sessions with six systems ("Bot 001" to "Bot 006").

We used dialogue sessions that satisfied the following criteria:

- There are 11 or more system utterances in a dialogue.
- There is no utterance that consists of only emoji.
- Utterances of a dialogue do not contain instruction statements such as "Text is not given. Please try to type /end and /test to reset the state and get text."
- Utterances of a dialogue do not contain "/test."

If an utterance contained the command "/start," utterances below the command were used. All emojis were replaced with text that represented their meanings, such as "[WAVING HAND SIGN]."

Dialogue breakdown annotation was performed by the Amazon Mechanical Turk (AMT)[2] crowd-sourcing service by using 15 annotators for each system utterance. In previous challenges, we used 24–30 annotators; however, we decreased the number of annotators to 15 in this challenge (except for the dialogue live competition data described below) for budget reasons and because we wanted to increase the number

[1] https://github.com/DeepPavlov/convai/tree/master/data.

[2] https://requester.mturk.com.

of dialogues for deep learning methods to be successfully applied. We informed the annotators that the task required native English skills.

4.1.1 Development and Evaluation Data

For the development data, 211 dialogue sessions were chosen at random from the mass of IRIS and ConvAI2 dialogues that satisfied our requirements. For the evaluation data, 200 dialogue sessions were also chosen at random. Each system utterance (except for the initial system prompt) was annotated by 15 annotators.

Table 1 shows the distribution and inter-annotator agreement (Fleiss' kappa) of the dialogue breakdown labels over the systems.

4.2 Datasets for Japanese

For the Japanese datasets, we asked the annotators to use the datasets we previously created for DBDC1–DBDC3. We also created and distributed new dialogues collected from the dialogue system live competition held in Japan in November 2018 [8]. We annotated the dialogues of five systems whose developers gave us consent to annotate and distribute the dialogues for public use. Here, we briefly describe the

Table 1 Statistics of English datasets

	Development data						
	Bot001	Bot002	Bot003	Bot004	Bot005	Bot006	IRIS
No. of sessions	39	38	42	41	2	6	43
No. of annotators	15	15	15	15	15	15	15
NB (Not a breakdown) (%)	40.4	40.8	35.8	39.9	22.0	16.4	30.0
PB (Possible breakdown) (%)	29.4	26.8	29.5	29.4	37.0	22.6	30.3
B (Breakdown) (%)	30.2	32.4	34.7	30.7	41.0	61.0	39.6
Fleiss' κ (NB, PB, B)	0.11	0.20	0.16	0.18	0.11	0.10	0.12
Fleiss' κ (NB, PB+B)	0.16	0.30	0.25	0.26	0.12	0.13	0.15
	Evaluation data						
	Bot001	Bot002	Bot003	Bot004	Bot005	Bot006	IRIS
No. of sessions	46	33	47	38	2	7	27
No. of annotators	15	15	15	15	15	15	15
NB (Not a breakdown) (%)	45.1	44.9	41.5	46.4	22.0	27.2	35.9
PB (Possible breakdown) (%)	31.3	29.3	31.4	29.5	19.3	33.0	29.7
B (Breakdown) (%)	23.6	25.8	27.0	24.2	58.7	39.8	34.4
Fleiss' κ (NB, PB, B)	0.09	0.12	0.14	0.12	0.09	0.06	0.08
Fleiss' κ (NB, PB+B)	0.13	0.18	0.20	0.17	0.15	0.11	0.12

Table 2 Statistics of Japanese datasets

	JCDC-1146		DBDC1	DBDC2 (DVL/EVL)			DBDC3 (EVL)		
	init100	rest1046	DVL/EVL	DCM	DIT	IRS	DCM	DIT	IRS
No. of sessions	100	1046	20/80	50/50	50/50	50/50	50	50	50
No. of annotators	24	2 or 3	30	30	30	30	30	30	30
NB (Not a Breakdown)	59.2%	58.3%	37.1%	39.8%	33.0%	37.4%	34.9%	25.3%	29.3%
PB (Possibly Breakdown)	22.2%	25.3%	32.2%	30.2%	27.4%	24.3%	34.2%	28.3%	23.8%
B (Breakdown)	18.6%	16.4%	30.6%	29.9%	39.5%	38.3%	30.9%	46.4%	46.9%
Fleiss' κ (NB, PB, B)	0.28	0.28	0.20	0.31	0.24	0.36	0.24	0.14	0.27
Fleiss' κ (NB, PB+B)	0.40	0.40	0.27	0.44	0.38	0.48	0.32	0.20	0.37

	Live Competition (DVL/EVL)					DBDC4 (EVL)		
	MMK	MRK	TRF	ZNK	IRS	DCM	DIT	IRS
No. of sessions	15/14	15/14	14/16	16/14	13/15	50	50	50
No. of annotators	30	30	30	30	30	15	15	15
NB (Not a Breakdown)	61.0%	48.9%	65.7%	42.7%	34.6%	47.7%	34.0%	38.8%
PB (Possibly Breakdown)	27.4%	34.0%	22.4%	31.4%	28.7%	31.6%	34.3%	29.2%
B (Breakdown)	11.6%	17.1%	12.0%	26.0%	36.7%	20.7%	31.8%	32.0%
Fleiss' κ (NB, PB, B)	0.11	0.15	0.19	0.21	0.29	0.22	0.13	0.29
Fleiss' κ (NB, PB+B)	0.16	0.21	0.26	0.29	0.37	0.28	0.17	0.40

development data and the newly created evaluation data. See Table 2 for the statistics of the Japanese datasets.

4.2.1 Development Data

Chat dialogue corpus This dataset has 1,146 dialogue sessions. The dialogues were collected using NTT DOCOMO's chat API (DCM) [10]. One hundred dialogues (called "init100") were annotated by 24 annotators; the rest of the dialogues (called "rest1046") were annotated by 2 or 3 annotators. Dialogue breakdown annotation was done by the researchers working on chat-oriented dialogue systems in Japan.

Development data for DBDC1 This dataset has 20 dialogue sessions. The dialogues were collected using DCM and the CrowdWorks[3] crowd-sourcing service and were annotated by 30 annotators using the Yahoo! Crowd-sourcing[4] service. All datasets in DBDC1–DBDC3 were collected and annotated in the same way.

Evaluation data for DBDC1 This dataset has 80 dialogue sessions. The dialogues were collected using DCM. Each system utterance was annotated by 30 annotators.

Development data for DBDC2 This dataset has 150 dialogue sessions. The dialogues were collected using DCM, DIT (Denso IT Laboratories' system) [14], and IRS (IR-status from [11]) systems. Each system utterance was annotated by 30 annotators.

[3]http://crowdworks.jp.

[4]http://crowdsourcing.yahoo.co.jp.

Evaluation data for DBDC2 This dataset has 150 dialogue sessions, 50 dialogues each from DCM, DIT, and IRS. Each system utterance was annotated by 30 annotators.

Evaluation data for DBDC3 This dataset has 150 dialogue sessions, 50 dialogues each from DCM, DIT, and IRS. Each system utterance was annotated by 30 annotators.

Dialogue system live competition data This dataset contains 73 dialogues from five systems. We had 146 dialogues as original data and split them to use half for development data and the other half for evaluation. Since there were only about 15 dialogues for each system, we used this dataset to train and test dialogue breakdown detectors when only a small amount of data was available. Each system utterance was annotated by 30 annotators.

4.2.2 Evaluation Data

The evaluation data for Japanese contained 223 dialogue sessions: 50 dialogues each from DCM, DIT, and IRS and 73 dialogues from the dialogue system live competition. This data was newly created for this challenge. Each system utterance for DCM, DIT, and IRS was annotated by 15 annotators. The dialogue system live competition data was annotated by 30 annotators. We used the same procedure that we used to create the evaluation data for DBDC2 and DBDC3 except for the number of annotators.

5 Submitted Runs

Out of ten teams registered, four teams submitted the results. Table 3 shows the descriptions of the systems for English, information about the teams, and descriptions of the runs. The table shows the information of the two baselines we prepared, one based on conditional random fields and one that simply uses the majority label in the training data. Notably, except for a run of RSL19BD, all runs used deep learning methods.

For the submitted runs for Japanese, two teams that submitted runs for English participated with the same methods used for English. See Table 4 for the descriptions of the teams and their runs and information about the baselines.

6 Results

Tables 5, 6, 7, 8, 9 and 10 show the results for all the metrics in the English and Japanese runs. The runs are sorted by their performance. The results are the average of all utterances in the datasets for each language.

Table 3 Description of systems for English

Team	Organization	# of runs	Method	Description of runs
LIIR	KU Leuven	1	LSTM	**run1:** Vanilla LSTM with GloVe Common Crawl embedding
NTTCS19	NTT Communication Science Labs.	4	BERT	We utilized BERT for dialogue breakdown detection with the additional features, dialogue act, sentence length, number of elapsed turns, and sentence similarities. **run1:** with DBDC4-dev + Metric:MSE; **run2:** with DBDC4-dev + Metric:Accuracy; **run3:** without DBDC4-dev + Metric:MSE; **run4:** without DBDC4-dev + Metric:Accuracy
BitTalk	Samsung Research America	1	MemNN, BiLSTM	**run1:** we exploited memory-like networks that consist of bidirectional LSTMs (BiLSTMs) and self-attention layers to incorporate dialogue context in representation learning. Note that only 211 English training sessions were used for building a model, although the organizers did not put any restriction on how much data can be used
RSL19BD	Waseda University	5	Random Forest Regressor, LSTM	**run1:** RandomForestRegressor; **run2:** LSTM-based model with CNN text feature extraction; **run3:** ensemble of 5 LSTM-based models; **run4:** ensemble of run 1 and run 2; **run5:** ensemble of run 1 and run 3
Baseline	Organizer	1	CRF	A conditional random field (CRF) based baseline that labels utterance sequences with the three breakdown labels by using CRFs. Words in a target utterance and its previous utterances are used as features
Baseline	Organizer	1	Majority	A majority baseline that output the most frequent dialogue breakdown label in each system's development data

Table 4 Description of systems for Japanese

Team	Organization	# of runs	Method	Description of runs
NTTCS19	NTT Communication Science Labs.	4	BERT	We utilized BERT for dialogue breakdown detection with the additional features, dialogue act, sentence length, number of elapsed turns, and sentence similarities. **run1:** with DBDC4-dev + Metric:MSE; **run2:** with DBDC4-dev + Metric:Accuracy; **run3:** without DBDC4-dev + Metric:MSE; **run4:** without DBDC4-dev + Metric:Accuracy
RSL19BD	Waseda University	5	Random Forest Regressor, LSTM	**run1:** RandomForestRegressor; **run2:** LSTM-based model with CNN text feature extraction; **run3:** ensemble of 5 LSTM-based models; **run4:** ensemble of run 1 and run 2; **run5:** ensemble of run 1 and run 3
Baseline	Organizer	1	CRF	A conditional random field (CRF) based baseline that labeled utterance sequences with the three breakdown labels by using CRFs. Words in a target utterance and its previous utterances were used as features
Baseline	Organizer	1	Majority	A majority baseline that output the most frequent dialogue breakdown label in each system's development data

Table 5 Overall results of JS divergence (English)

Run	JSD (NB, PB, B)	Run	JSD (NB, PB+B)	Run	JSD (NB+PB, B)
RSL19BD run5	0.0662	RSL19BD run5	0.0389	RSL19BD run5	0.0416
RSL19BD run3	0.0675	NTTCS19 run1	0.0389	RSL19BD run3	0.0424
RSL19BD run4	0.069	NTTCS19 run2	0.0391	NTTCS19 run4	0.0433
NTTCS19 run4	0.0693	RSL19BD run3	0.0401	RSL19BD run4	0.0438
RSL19BD run1	0.07	NTTCS19 run4	0.0407	RSL19BD run1	0.0438
NTTCS19 run3	0.0709	RSL19BD run4	0.0412	NTTCS19 run3	0.0449
RSL19BD run2	0.0725	NTTCS19 run3	0.0417	RSL19BD run2	0.0462
NTTCS19 run1	0.0733	RSL19BD run1	0.042	NTTCS19 run2	0.0481
NTTCS19 run2	0.0752	RSL19BD run2	0.0439	NTTCS19 run1	0.0504
majority run1	0.08	Majority run1	0.0507	Majority run1	0.0505
BitTalk run1	0.0992	BitTalk run1	0.0706	BitTalk run1	0.057
LIIR run1	0.4245	Baseline run1	0.3176	Baseline run1	0.267
baseline run1	0.4381	LIIR run1	0.4193	LIIR run1	0.4186

Table 6 Overall results of mean squared error (English)

Run	MSE (NB, PB, B)	Run	MSE (NB, PB+B)	MSE	MSE (NB+PB, B)
RSL19BD run5	0.0336	NTTCS19 run2	0.0432	RSL19BD run5	0.0369
RSL19BD run3	0.0346	RSL19BD run5	0.0439	RSL19BD run3	0.0381
NTTCS19 run4	0.0351	NTTCS19 run1	0.0444	NTTCS19 run4	0.0384
RSL19BD run4	0.0353	RSL19BD run3	0.0455	RSL19BD run4	0.0389
NTTCS19 run3	0.0361	NTTCS19 run4	0.0455	RSL19BD run1	0.0398
RSL19BD run1	0.0362	RSL19BD run4	0.0469	NTTCS19 run3	0.0399
RSL19BD run2	0.0374	NTTCS19 run3	0.0472	RSL19BD run2	0.0414
NTTCS19 run2	0.0377	RSL19BD run1	0.048	NTTCS19 run2	0.0429
NTTCS19 run1	0.0378	RSL19BD run2	0.0506	Majority run1	0.0449
Majority run1	0.0415	Majority run1	0.0583	NTTCS19 run1	0.0465
BitTalk run1	0.0506	BitTalk run1	0.0734	BitTalk run1	0.0513
LIIR run1	0.1323	LIIR run1	0.2641	Baseline run1	0.2295
Baseline run1	0.2237	Baseline run1	0.2788	LIIR run1	0.2853

As we noted, we focused on the results of MSE(NB, PB, B) and MSE(NB+PB, B), which were more reliable than other metrics (see [15]). As for English, RSL19BD run5 with 0.0336 of MSE(NB, PB, B) and 0.0369 of MSE(NB+PB, B) performed the best. The method used was based on the combination of a Random Forest Regressor and an LSTM-based method. As for Japanese, NTTCS19 run1 with 0.0463 of MSE(NB, PB, B) and 0.0507 of MSE(NB+PB, B) performed the best. The method used was based on Bidirectional Encoder Representations from Transformers (BERT) [2]. For the details of the methods, please refer to their system description papers [4, 12, 13, 16].

We analyzed the performance of the upper bound by splitting the annotators, regarding one half as references and the other half as hypotheses. Refer to Tables 11 and 12. Regarding MSE(NB, PB, B) and MSE(NB+PB, B), the upper bounds were

Table 7 Overall results of classification (English)

Run	Accuracy	Run	F1 (B)	Run	F1 (PB+B)
NTTCS19 run4	0.556	RSL19BD run5	0.469	NTTCS19 run1	0.7664
Majority run1	0.5365	RSL19BD run4	0.465	RSL19BD run2	0.7276
NTTCS19 run3	0.5345	NTTCS19 run1	0.4641	RSL19BD run4	0.7174
NTTCS19 run2	0.532	RSL19BD run3	0.4554	RSL19BD run3	0.6961
LIIR run1	0.5335	NTTCS19 run3	0.4547	RSL19BD run5	0.6947
RSL19BD run5	0.5255	RSL19BD run2	0.4483	RSL19BD run1	0.674
RSL19BD run3	0.52	NTTCS19 run2	0.4482	NTTCS19 run3	0.6724
RSL19BD run4	0.505	RSL19BD run1	0.4411	BitTalk run1	0.6492
RSL19BD run1	0.499	NTTCS19 run4	0.4403	NTTCS19 run2	0.6369
NTTCS19 run1	0.488	BitTalk run1	0.3901	NTTCS19 run4	0.6079
RSL19BD run2	0.473	Baseline run1	0.3421	Baseline run1	0.5803
Baseline run1	0.4635	LIIR run1	0.0981	LIIR run1	0.0984
BitTalk run1	0.4355	Majority run1	0	Majority run1	0

Table 8 Overall results of JS divergence (Japanese)

Run	JSD (NB, PB, B)	Run	JSD (NB, PB+B)	Run	JSD (NB+PB, B)
RSL19BD run5	0.0947	RSL19BD run5	0.0601	NTTCS19 run1	0.0612
NTTCS19 run1	0.0953	RSL19BD run4	0.0602	RSL19BD run5	0.0636
RSL19BD run4	0.0954	RSL19BD run3	0.0615	RSL19BD run4	0.0646
RSL19BD run3	0.0967	NTTCS19 run1	0.062	RSL19BD run1	0.0647
RSL19BD run1	0.0975	RSL19BD run2	0.0623	RSL19BD run3	0.0659
RSL19BD run2	0.0989	RSL19BD run1	0.0627	NTTCS19 run2	0.0674
NTTCS19 run2	0.1014	NTTCS19 run2	0.0642	RSL19BD run2	0.0684
NTTCS19 run3	0.1259	NTTCS19 run3	0.084	Majority run1	0.0818
Majority run1	0.136	NTTCS19 run4	0.0937	NTTCS19 run3	0.0871
NTTCS19 run4	0.1361	Majority run1	0.1016	NTTCS19 run4	0.0933
Baseline run1	0.3839	Baseline run1	0.2628	Baseline run1	0.2342

Table 9 Overall results of mean squared error (Japanese)

Run	MSE (NB, PB, B)	Run	MSE (NB, PB+B)	Run	MSE (NB+PB, B)
NTTCS19 run1	0.0463	NTTCS19 run1	0.0635	NTTCS19 run1	0.0507
RSL19BD run5	0.0475	RSL19BD run5	0.064	RSL19BD run5	0.0556
RSL19BD run4	0.048	RSL19BD run4	0.0643	RSL19BD run1	0.0568
RSL19BD run1	0.0492	RSL19BD run3	0.0662	RSL19BD run4	0.0568
RSL19BD run3	0.0493	NTTCS19 run2	0.067	NTTCS19 run2	0.0581
NTTCS19 run2	0.0504	RSL19BD run1	0.0671	RSL19BD run3	0.0591
RSL19BD run2	0.0509	RSL19BD run2	0.0674	RSL19BD run2	0.0622
NTTCS19 run3	0.0653	NTTCS19 run3	0.0915	Majority run1	0.0669
Majority run1	0.0669	Majority run1	0.1026	NTTCS19 run3	0.0779
NTTCS19 run4	0.0715	NTTCS19 run4	0.1028	NTTCS19 run4	0.0848
Baseline run1	0.1997	Baseline run1	0.2293	Baseline run1	0.2085

Table 10 Overall results of classification (Japanese)

Run	Accuracy	Run	F1 (B)	Run	F1 (PB+B)
NTTCS19 run1	0.5841	NTTCS19 run1	0.5387	NTTCS19 run2	0.7254
NTTCS19 run2	0.5724	NTTCS19 run2	0.5255	NTTCS19 run1	0.7091
RSL19BD run3	0.5476	RSL19BD run2	0.4613	NTTCS19 run4	0.7026
RSL19BD run5	0.5444	NTTCS19 run3	0.4605	RSL19BD run2	0.7014
RSL19BD run2	0.5412	RSL19BD run5	0.4603	NTTCS19 run3	0.7011
RSL19BD run4	0.5412	RSL19BD run3	0.4589	RSL19BD run3	0.6811
RSL19BD run1	0.539	RSL19BD run4	0.4583	RSL19BD run4	0.6782
Baseline run1	0.533	RSL19BD run1	0.4568	RSL19BD run5	0.6667
Majority run1	0.4993	NTTCS19 run4	0.4512	Baseline run1	0.6592
NTTCS19 run3	0.4808	Baseline run1	0.4367	RSL19BD run1	0.656
NTTCS19 run4	0.4446	Majority run1	0.3846	Majority run1	0.5485

Table 11 Upper bound (classification-related metrics)

	Accuracy	Precision(B)	Recall(B)	F-measure(B)
English	0.5891	0.4971	0.5284	0.5116
Japanese	0.7884	0.7296	0.7328	0.7305
	Precision(PB+B)	Recall(PB+B)	F-measure(PB+B)	
English	0.7809	0.6808	0.7272	
Japanese	0.9028	0.7396	0.8128	

Table 12 Upper bound (distribution-related metrics)

	JSD(NB, PB, B)	JSD(NB, PB+B)	JSD(NB+PB, B)
English	0.1169	0.0577	0.0559
Japanese	0.0495	0.0254	0.0239
	MSE(NB, PB, B)	MES(NB, PB+B)	MSE(NB+PB, B)
English	0.0516	0.0541	0.0459
Japanese	0.0196	0.0215	0.0142

0.0516 and 0.0459 for English and 0.0196 and 0.0142 for Japanese, indicating that that the English runs achieved a closer value to human-level performance while there is still room to improve for Japanese. We attribute this to the rather low inter-agreement for the English datasets; high variance in the annotations can make the upper bound lower. When we looked at the performance of F1(B) for Japanese, we saw there is a lot of room for improvement, given the best run achieved 53% whereas the human performance is about 73%. Although we found MSE metrics to be more reliable than those in our previous study, we may need to reconsider our preferences regarding the metrics in terms of how much room there is compared to the upper bounds.

7 Summary and Future Work

We overviewed our Dialogue Breakdown Detection Challenge 4 (DBDC4) for which we prepared both English and Japanese datasets. Out of the ten teams registered, four teams participated using various methods of detecting dialogue breakdown. Most of the submitted runs used deep-learning methods and achieved promising results. We obtained accuracies close to the upper bound in English, but there is still room for improvement for Japanese, likely due to the low inter-annotator agreement in the English dataset. We plan to consider ways to improve the quality of the dataset.

Acknowledgements We would like to thank the participants for their efforts to explore new methods and for submitting their runs and system description papers. We thank Rafael E. Banchs and members of the Conversational Intelligence Challenge for graciously providing us with datasets to make DBDC4 possible. We also thank our sponsors, Denso IT Laboratories and National University of Singapore, for supporting our data collection.

References

1. Banchs RE, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In: Proceedings of the ACL 2012 system demonstrations, pp 37–42
2. Devlin J, Chang MW, Lee K, Toutanova K (2018) BERT: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805
3. D'Haro LF, Shawar BA, Yu Z (2016) RE-WOCHAT 2016—shared task description report. In: Proceedings of the workshop on collecting and generating resources for chatbots and conversational agents-development and evaluation (RE-WOCHAT), pp 39–42
4. Hendriksen M, Leeuwenberg A, Moens MF (2019) LSTM for dialogue breakdown detection: exploration of different model types and word embeddings. In: Proceedings of the WOCHAT
5. Higashinaka R, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y, Mizukami M (2015) Towards taxonomy of errors in chat-oriented dialogue systems. In: Proceedings of the SIGDIAL, pp 87–95
6. Higashinaka R, Mizukami M, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y (2015) Fatal or not? Finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In: Proceedings of the EMNLP, pp 2243–2248
7. Higashinaka R, Funakoshi K, Kobayashi Y, Inaba M (2016) The dialogue breakdown detection challenge: task description, datasets, and evaluation metrics. In: Proceedings of the LREC, pp 3146–3150
8. Higashinaka R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Akama R (2019) Dialogue system live competition: identifying problems with dialogue systems through live event. In: Proceedings of the IWSDS
9. Martinovsky B, Traum D (2003) The error is the clue: breakdown in human-machine interaction. In: Proceedings of the ISCA workshop on error handling in spoken dialogue systems, pp 11–16
10. Onishi K, Yoshimura T (2014) Casual conversation technology achieving natural dialog with computers. NTT DOCOMO Tech J 15(4):16–21
11. Ritter A, Cherry C, Dolan WB (2011) Data-driven response generation in social media. In: Proceedings of the EMNLP, pp 583–593
12. Shin J, Dirafzoon A, Anshu A (2019) Context-enriched attentive memory network with global and local encoding for dialogue breakdown detection. In: Proceedings of the WOCHAT

13. Sugiyama H (2019) Dialogue breakdown detection using BERT with traditional dialogue features. In: Proceedings of the WOCHAT
14. Tsukahara H, Uchiumi K (2015) System utterance generation by label propagation over association graph of words and utterance patterns for open-domain dialogue systems. In: Proceedings of the PACLIC, pp 323–331
15. Tsunomori Y, Higashinaka R, Takahashi T, Inaba M (2018) Evaluating dialogue breakdown detection in chat-oriented dialogue systems. In: Proceedings of the SEMDIAL
16. Wang C, Kato S, Sakai T (2019) RSL19BD at DBDC4: ensemble of decision tree-based and LSTM-based models. In: Proceedings of the WOCHAT

Dialogue Breakdown Detection Using BERT with Traditional Dialogue Features

Hiroaki Sugiyama

Abstract Despite of the significant improvements of Natural Language Processing with Neural networks such as machine reading comprehensions, chat-oriented dialogue systems sometimes generate inappropriate response utterances that cause dialogue breakdown because of the difficulty of generating utterances. If we can detect such inappropriate utterances and suppress them, dialogue systems can continue the dialogue easily.

1 Introduction

Chatting with people is an important function of dialogue systems in building social relationships with users. This not only provides therapeutic and entertainment benefits but also plays an important role in drawing out the user's potential requirements and constructing a good relationship with the user. Furthermore, such conversational dialogue has the potential to improve the performance of task-oriented dialogue [1]. Thus, the construction of conversational dialogue systems (also called non-task oriented dialogue systems or chat-oriented dialogue systems) has recently gained attention [2–4].

The difficulty of developing chat-oriented dialogue systems is that such systems are required to respond to a very wide range of topics expressed by user utterances. Since it is still difficult for the current dialogue systems to continue outputting appropriate responses, utterances that cause the dialogue to collapse are often generated. It is assumed that the continuation of dialogue becomes easy when we can detect and suppress such problematic utterances.

In a previous dialogue breakdown detection challenge (DBDC), the author proposed a dialogue breakdown detection system that captures frequently appearing error patterns that are specific to each utterance generation approach [5]. The authors

H. Sugiyama (✉)
NTT Communication Science Laboratories, Kyoto, Japan
e-mail: h.sugi@ieee.org

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_39

proved that traditional dialogue features such as dialogue-acts or sentence similarities calculated with word vectors are effective to such errors.

From the other viewpoint, machine reading comprehension with neural networks are so popular recently, and many significant improvements are frequently proposed. Especially, BERT achieves SOTA in many natural language processing tasks and has gained attention. BERT adopts pre-training task called Next Sentence Prediction (NSP) that evaluates the cohesion of sentence pairs. Since NSP task resembles dialogue breakdown detection, we expect that BERT improves dialogue breakdown detection performance. In this paper, we propose a novel dialogue breakdown detection method that combines BERT and traditional dialogue features, and examine the effectiveness of the additional features.

2 Systems

We utilize BERT [6] to detect dialogue breakdown. In addition to the original BERT, we introduce traditional dialogue features such as dialogue-act to improve the estimation performance. In this section, we explain the structure of our model and the details of additional features.

2.1 BERT with Additional Features

BERT is a Transformer-based method that achieves SOTA performances in many kinds of Natural Language Processing tasks [6]. The important advantage of BERT is that, once a transformer model is pre-trained with large-scale text corpus, we can fine-tune the pre-trained model with fewer data than the scratch.

When we adopt BERT to classification tasks, we use the final hidden state of Transformer corresponding to first token ([CLS]) as the aggregate sequence representation, to which we adopt feed-forward networks and softmax functions for final outputs (classification results). The feed-forward network is pre-trained with Next Sentence Prediction (NSP) task, where BERT predicts whether randomly chosen two sentences A and B are actually connected (A follows B) or not. Since this NSP task resembles our dialogue breakdown task, we expect that BERT improves the estimation performance. However, since BERT pre-trained models in public are not trained with dialogue corpus, the prediction possibly does not suit for our task. Besides, pre-training of the model with dialogue data is difficult because it requires a huge size of texts.

To overcome this difficulty, we concatenate additional dialogue features that represent the naturalness of dialogue flow to the aggregate representation. Figure 1 illustrates our model. Original BERT estimates only aggregate representation C in the left side. We utilize word (token) vectors T obtained from BERT to calculate

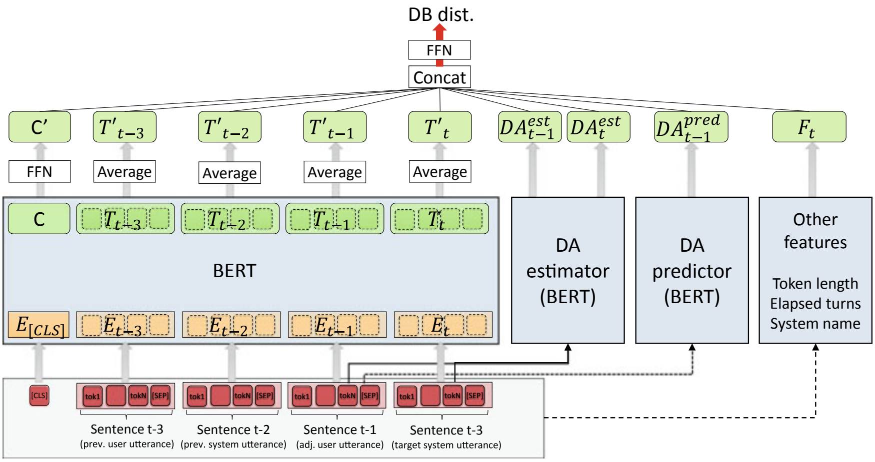


Fig. 1 Distribution of annotated winning rates between annotators

sentence vectors of the dialogue context. Dialogue acts are separately estimated and predicted using other BERT models, and are concatenated to the vector C and T.

2.2 *Features*

In this section, we explain the details of the dialogue features shown in Fig. 1.

2.2.1 Dialogue Act

Dialogue act represents the action types of each utterance such as *question* or *greeting*. This is a very common dialogue feature to represent dialogue flow [7], and is proved as an effective feature for dialogue breakdown detection [5]. For example, in DBDC, all of the systems sometimes respond with questions even when the user utterance is a question. Dialogue acts are useful to capture such kinds of errors. In this study, we utilize *estimated dialogue acts* of the target system utterance and adjacent user utterance, and in addition, we use *predicted dialogue acts* that are expected to be suitable for the next utterance after the user utterance. We use a dialogue acts definition proposed in [8], in which they categorize utterances into 33 dialogue acts.

We train dialogue acts estimator and predictor with regression of multi-label expressions (e.g.., [0,0,1,0,1,0,..]), since one utterance possibly contains multiple dialogue acts.

We train Japanese dialogue acts estimator and predictor with NTT's Japanese chat dialogue corpus (3680 dialogues) [4], using BERT with words that are tokenized with sentencepiece. English dialogue acts estimator and predictor were trained with NTT's English situation dialogue corpus (4000 dialogues), using BERT with words.

2.2.2 Sentence Length

With the technology of the current dialogue system, it is difficult to estimate the consistency of the user utterance and the system utterance. Therefore, there is the problem that the longer the system utterance is, the greater the possibility that an unrelated element is included. In particular, DIT system of Japanese task tends to generate very long utterances, and thus, the utterance does not match the content of the user utterance. Here, we add token length of the target utterance to the features.

2.2.3 Number of Elapsed Turns

All three dialogue systems generate relatively appropriate utterances at the beginning of a dialogue, but the proportion of inappropriate utterances tends to increase as the dialogue proceeds. Therefore, the number of elapsed turns from the start of the dialogue is added to the features.

2.2.4 Sentence Embedding and Similarities

DIT and IRS systems in Japanese task sometimes generate system utterances with topics completely different from user utterances. DCM system in Japanese task sticks to a specific topic, and, as a result, there are many cases where an utterance with almost the same contents repeatedly occurs. In order to detect these errors, we adopt word-based sentence embedding features, the difference vectors of the sentence embedding, and sentence similarity features between system and user utterances and between the target and a previous system utterance. Sentence embedding is calculated as the average vector of the last layer of BERT corresponding the tokens in target sentences.

2.2.5 System Names

It is shown in previous DBDC papers that each dialogue system has a specific dialogue breakdown patterns [5]. To capture the system-dependent error patterns, we adopt a system name feature with one-of-k vector representation.

Table 1 Dataset distributed in DBDC English task [11]

	DBDC3 (eval)				DBDC4 (dev/eval)						
	CIC	TKTK	YI	IRIS	Bot001	Bot002	Bot003	Bot004	Bot005	Bot006	IRIS
No. of dialogues	100/50	100/50	100/50	100/50	39/46	38/33	42/47	41/38	2/2	6/7	43/27
No. of annotators	30	30	30	30	15	15	15	150	15	15	15

Table 2 Dataset distributed in DBDC Japanese task [11]

	DBDC1 (dev/eval)	DBDC2 (dev/eval)			DBDC3 (eval)			DBDC4 (dev/eval)			
	DCM	DCM	DIT	IRS	DCM	DIT	IRS	DCM	DIT	IRS	LiveComp
No. of dialogues	20/80	50/50	50/50	50/50	50	50	50	0/50	0/50	0/50	73/73
No. of annotators	30	30	30	30	30	30	30	15	15	15	30

3 Experiment

3.1 Dataset

3.1.1 Dataset for English Task

The DBDC organizers have distributed previous DBDC3 dataset as development dataset for English task [9]. Table 1 shows the statistics of the dataset. The target dialogue systems of DBDC4 are dialogue systems submitted for Conversational Intelligence Challgenge 2 (ConvAI2)[1] and IRIS [10], which are changed from those of DBDC3 (CIC, TKTK, IRIS and YI).

3.2 Dataset for Japanese Task

The DBDC organizers have distributed development dataset for Japanese task using previous DBDC1, DBDC2 and DBDC3 [9]. Table 2 shows the statistics of the dataset. DBDC4 task contains a new dialogue system group called LiveComp, which consists of four dialogue systems developed for Dialogue system live competition [12].

3.3 Experiment Settings

In this research, we examined the effectiveness of the features described in Sect. 2.2 by adding certain features to the case of original BERT (no additional features).

[1] https://github.com/DeepPavlov/convai/tree/master/data.

Table 3 Feature addition analysis in DBDC English task

	DBDC3-eval		
	Accuracy	MSE	JSD
Original BERT	0.5180	0.0231	0.0410
+Dialogue act(DA)	0.5250	0.0227	0.0409
+Sentence vector(SV)	0.5375	0.0225	0.0408
+SV +Sentence distance(Dis)	0.5055	0.0231	0.0408
+SV +Difference of SV(Diff)	0.5205	0.0226	0.0405
+System names(Sys)	0.5165	0.0225	0.0405
+Elapsed turns and sentence length(Other)	**0.5375**	0.0227	0.0405
+DA +SV	0.5265	0.0225	0.0403
+DA +SV +Dis +Diff	0.5175	0.0227	0.0406
+DA +SV +Dis +Diff +Sys + Other(ALL)	0.5315	**0.0222**	**0.0399**

Since importance is placed on distribution-related metrics (Mean Squared Error and JS divergence), we examined the features that minimize MSE.

We used DBDC1-dev, DBDC1-eval and DBDC2-dev, DBDC2-eval for the training data in Japanese task, and DBDC3-dev in English task. DBDC3-eval is used for evaluation data in both tasks. DBDC4-dev data is distributed but its size is very small. Since DBDC4-dev data shows similar behaviors of the performance with DBDC3-dev, we add DBDC4-dev to train data instead of using for validation data.

because this analysis adopted MSE as model selection and optimization function.

Each model with a certain feature is trained using Adabound optimization method [13]. For choosing models submitted to DBDC4 shown in Sect. 4, we use optuna[2] developed by Preferred Networks to search hyper-parameters (final lr and training batch size) and optimum feature sets. However, after the submission, we noticed that random seed is more dominant to find out the best performance of a certain feature set. Therefore, for the analysis shown in Sect. 3.4, we examine the feature sets with the following three parameter tuning steps. First, we tune *final lr* of Adabound with optuna. Second, using the optimum *final lr*, we search for better random seed randomly. Finally, using the best random seed, we tune *final lr* again around the firstly chosen value.

3.4 Result

Tables 3 and 4 show the result of the feature addition analysis of English and Japanese tasks. They illustrate that models with all the features achieved the best performance

[2] https://optuna.org/.

Table 4 Feature addition analysis in DBDC Japanese task

	DBDC3-eval		
	Accuracy	MSE	JSD
Original BERT	0.6018	0.0401	0.0765
+Dialogue act(DA)	0.5885	0.0400	**0.0757**
+Sentence vector(SV)	0.6012	0.0398	0.0762
+SV +Sentence distance(Dis)	0.6000	0.0397	0.0758
+SV +Difference of SV(Diff)	**0.6030**	0.0397	0.0758
+System names(Sys)	0.5982	0.0406	0.0781
+Elapsed turns and sentence length(Other)	**0.6030**	0.0404	0.0768
+DA +SV	0.6018	0.0399	0.0760
+DA +SV +Dis +Diff	0.6012	0.0399	0.0760
+DA +SV +Dis +Diff +Sys + Other(ALL)	0.5964	**0.0395**	0.0759

in MSE both English and Japanese tasks, and the models are superior to the original BERT. In the comparison of each feature, sentence vectors(SV) seems effective in both tasks.

On the other hand, Accuracy and JSD metrics are not consistent with MSE result,

4 Submitted Systems

We adopt models trained with all the features for submitted systems. We prepare four systems for each language with the combination of their training data (the use of DBDC4-dev), and metrics used for model selection (highest accuracy or lowest MSE). In addition, as extra trials after the competition, we evaluate another run that utilizes only DBDC4-dev for training.

Table 5 shows the result of DBDC4 English task. Run 4, which is trained only with DBDC3-dev and is selected as the best accuracy model, shows the best performance

Table 5 Submitted systems and extended trials for DBDC4-en

Runs	DBDC3-dev	DBDC4-dev	Metric	DBDC3-eval			DBDC4-eval		
				Acc	MSE	JSD	Acc	MSE	JSD
Run 1	✓	✓	MSE	0.521	0.0229	0.0410	0.488	0.0378	0.0732
Run 2	✓	✓	Accuracy	0.521	0.0230	0.0439	0.532	0.0376	0.0752
Run 3	✓	–	MSE	0.525	0.0223	0.0404	0.534	0.0360	0.0708
Run 4	✓	-	Accuracy	0.538	0.0225	0.0409	**0.556**	**0.0350**	**0.0692**
*EX 1	–	✓	MSE(RUN 3)	0.547	0.0233	0.0412	**0.601**	**0.0299**	**0.0580**

Table 6 Submitted systems and extended trials for DBDC4-ja

Runs	DBDC1,2-dev/eval	DBDC4-dev	Metric	DBDC3-eval			DBDC4-eval		
				Acc	MSE	JSD	Acc	MSE	JSD
Run 1	✓	✓	MSE	0.587	0.0414	0.0801	**0.584**	**0.0462**	**0.0953**
Run 2	✓	✓	Accuracy	0.605	0.0427	0.0809	0.572	0.0504	0.1013
Run 3	✓	–	MSE	0.598	0.0406	0.0779	0.480	0.0653	0.1259
Run 4	✓	–	Accuracy	0.605	0.0414	0.0795	0.444	0.0714	0.1360
*EX 1	–	✓	MSE(Run 3)	0.596	0.0421	0.0794	**0.599**	**0.0451**	**0.0763**

among Run 1–4. Although the English task of DBDC4-eval does not contain DBDC3 systems, Run 1 and 2 are lower performance than Run 3 and 4. Considering the model EX 1 trained only with DBDC4-dev is superior to the other settings (including other teams), DBDC4-dev is crucial for the training. This indicates that the dialogue systems' behaviors of DBDC3-eval resemble those of DBDC4-eval in the English task. We assume that the larger and more various training data of Run 1 and 2 make the Run 1 and 2 models search parameters more difficult than Run 3 and 4.

Table 6 illustrates the result of DBDC4 Japanese task. Run 1 trained with DBDC1,2-dev/eval and DBDC4-dev achieves the best score in Run 1–4. Run 3 and 4 trained only with DBDC1,2-dev/eval show significantly lower performance than Run 1 and 2. This indicates that DBDC4-dev is necessary to achieve high performance in DBDC4-eval.

More interestingly, EX 1 (extended trials) that leverages only DBDC4-dev for training shows the best result in all the runs including other teams. This shows that DBDC1,2-dev/eval is not necessary to achieve high performance in DBDC4-eval even though DBDC4-eval contains systems same as DBDC1,2.

5 Conclusion

We examined the effectiveness of additional features to improve the performance of BERT for dialogue breakdown detection. Through the analysis, sentence vectors are effective for the estimation but models trained with all the features show the best performance in both English and Japanese tasks. In addition, a comparison of training dataset shows that the training data domain is dominant for the performance. We plan to investigate the effectiveness of pre-training with huge dialogue data in English.

References

1. Bickmore T, Cassell J (2001) Relational agents: a model and implementation of building user trust. In: Proceedings of the SIGCHI conference on human factors in computing systems, pp 396–403
2. Ritter A, Cherry C, Dolan WB (2011) Data-driven response generation in social media. In: Proceedings of the 2011 conference on empirical methods in natural language processing, pp 583–593
3. Wong W, Cavedon L, Thangarajah J, Padgham L (2012) Strategies for mixed-initiative conversation management using question-answer pairs. In: Proceedings of the 24th international conference on computational linguistics, pp. 2821–2834
4. Higashinaka R, Imamura K, Meguro T, Miyazaki C, Kobayashi N, Sugiyama H, Hirano T, Makino T, Matsuo Y (2014) Towards an open-domain conversational system fully based on natural language processing. In: Proceedings of the 25th international conference on computational linguistics, pp 928–939
5. Sugiyama H (2017) Dialogue breakdown detection based on estimating appropriateness of topic transition. In: Proceedings of dialog system technology challenges, vol 6
6. Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the annual conference of the North American Chapter of the Association for Computational Linguistics
7. Meguro T, Higashinaka R, Dohsaka K, Minami Y, Isozaki H (2009) Analysis of listening-oriented dialogue for building listening agents. In: Proceedings of the SIGDIAL 2009 conference: the 10th annual meeting of the special interest group on discourse and dialogue (September), pp 124–127
8. Meguro T, Higashinaka R, Minami Y, Dohsaka K (2010) Controlling listening-oriented dialogue using partially observable markov decision processes. In: Proceedings of the 23rd international conference on computational linguistics, pp 761–769
9. Higashinaka R, D'Haro LF, Shawar BA, Banchs R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Sedoc Ja (2019) Overview of the dialogue breakdown detection challenge 4. In: Proceedings of WOCHAT
10. Banchs RE, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In: Proceedings of the 50th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp 37–42
11. Higashinaka R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Kaji N (2017) Overview of dialogue breakdown detection challenge 3. In: Proceedings of the dialogue system technology challenge, vol 6
12. Higashinaka R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Akama R (2019) Dialogue system live competition: identifying problems with dialogue systems through live event. In: Proceedings of international workshop on spoken dialogue systems technology
13. Luo L, Xiong Y, Liu Y, Sun X (2019) Adaptive gradient methods with dynamic bound of learning rate. In: Proceedings of the international conference on learning representations

RSL19BD at DBDC4: Ensemble of Decision Tree-Based and LSTM-Based Models

Chih-Hao Wang, Sosuke Kato, and Tetsuya Sakai

Abstract RSL19BD (Waseda University Sakai Laboratory) participated in the Fourth Dialogue Breakdown Detection Challenge (DBDC4) and submitted five runs to both English and Japanese subtasks. In these runs, we utilise the Decision Tree-based model and the Long Short-Term Memory-based (LSTM-based) model following the approaches of RSL17BD and KTH in the Third Dialogue Breakdown Detection Challenge (DBDC3) respectively. The Decision Tree-based model follows the approach of RSL17BD but utilises RandomForestRegressor instead of ExtraTreesRegressor. In addition, instead of predicting the mean and the variance of the probability distribution of the three breakdown labels, it predicts the probability of each label directly. The LSTM-based model follows the approach of KTH with some changes in the architecture and utilises Convolutional Neural Network (CNN) to perform text feature extraction. In addition, instead of targeting the single breakdown label and minimising the categorical cross entropy loss, it targets the probability distribution of the three breakdown labels and minimises the mean squared error. Run 1 utilises a Decision Tree-based model; Run 2 utilises an LSTM-based model; Run 3 performs an ensemble of 5 LSTM-based models; Run 4 performs an ensemble of Run 1 and Run 2; Run 5 performs an ensemble of Run 1 and Run 3. Run 5 statistically significantly outperformed all other runs in terms of MSE (NB, PB, B) for the English data and all other runs except Run 4 in terms of MSE (NB, PB, B) for the Japanese data (alpha level = 0.05).

C.-H. Wang (✉) · S. Kato · T. Sakai
Waseda University, Shinjuku-ku Okubo 3-4-1, Tokyo, Japan
e-mail: haohaowang.oscar@moegi.waseda.jp

S. Kato
e-mail: sow@suou.waseda.jp

T. Sakai
e-mail: tetsuyasakai@acm.org

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_40

1 Introduction

The task in the Fourth Dialogue Breakdown Detection Challenge (DBDC4) [3] is to build a model that detects whether an utterance from the system causes a breakdown in a dialogue context involving a system and a user. A breakdown is defined as a situation where the user cannot proceed with the conversation. Given a system utterance, the model is required to produce two outputs: 1. A single breakdown label chosen from the three breakdown labels (NB: Not a breakdown, PB: Possible breakdown, and B: Breakdown). 2. The probability distribution of the three breakdown labels, which we refer as P(NB), P(PB), and P(B) hereinafter. For evaluating the model, the organisers put an emphasis on the mean squared error (MSE). A complete description of the challenge can be found in the DBDC4 overview paper [3].

RSL19BD (Waseda University Sakai Laboratory) participated in DBDC4 and submitted five runs to both English and Japanese subtasks. In these runs, we utilise the Decision Tree-based model and the Long Short-Term Memory-based (LSTM-based) model following the approaches of RSL17BD [6] and KTH [8] in the Third Dialogue Breakdown Detection Challenge (DBDC3) [4] respectively.

2 Prior Art

At DBDC3 [4], RSL17BD [6] and KTH [8] both submitted models which achieved high performances. This section briefly describes their approaches.

2.1 RSL17BD at DBDC3

The top-performing model of RSL17BD utilises ExtraTreesRegressor [2][1] and employed the following features along with a few others based on pattern analysis: turn-index of the target utterance, length of the target utterance, and keyword flags of the target utterance. It first predicts the mean and variance of the probability distribution of the three breakdown labels for each target system utterance and then derives the predicted probability distribution. The single breakdown label is determined by choosing the label with the highest probability.

2.2 KTH at DBDC3

The top-performing model of KTH utilises Long Short-Term Memory (LSTM) [5]. For the preprocessing of English data, it produces a sequence of 300 dimensional

[1] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html.

word embedding vectors for each utterance in every dialogue and take the average sum of the sequence to produce the final embedding for a single utterance. This produces an embedded dialogue with each turn represented by a single 300 dimensional utterance embedding. The embedded dialogue is then processed by 4 LSTM layers and a Dense layer to produce 4 outputs for each turn. The 4 outputs are P(NB), P(PB), P(B), and P(U), where P(U) refers to the probability of user turn. The reason for adding P(U) is that user turns are included in the embedded dialogue as well and need to be predicted with a label different from NB, PB, and B. The model is trained for 100 epochs using Adadelta [13] as its optimiser. During training, it targets the single breakdown label and aims to minimise the categorical cross entropy loss for each target system utterance. For Japanese data, KTH did not submit any runs.

3 Model Descriptions

3.1 Decision Tree-Based Model

For the preprocessing of both English and Japanese data, we follow the same approach as RSL17BD [6] at DBDC3 [4]. Our model employs the same set of features as RSL17BD's model, but utilises RandomForestRegressor [1][2] instead of ExtraTreesRegressor [2]. In addition, instead of predicting the mean and the variance of the probability distribution over the three breakdown labels and then deriving the probability of each label, it predicts the probability of each label directly. The probability distribution is then calculated by normalising the probability of the three labels by their sum. The modifications above are decided by training and evaluating different model configurations, which can be found in the full version of this paper [12].

3.2 LSTM-Based Model

Following the approach of KTH [8] at DBDC3 [4], we utilise Long Short-Term Memory (LSTM) [5]. However, instead of taking the average sum of word embedding vectors for each utterance, we utilise Convolutional Neural Networks (CNN) to perform text feature extraction and produce the final embedded utterance. In addition, instead of targeting the single breakdown label and minimising the categorical cross entropy loss for each target system utterance, our model targets the probability distribution of the three breakdown labels and minimises its mean squared error. We chose Adam [7] as our optimiser and mean squared error as our loss function. Figure 1 shows the architecture diagram of our model.

[2]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

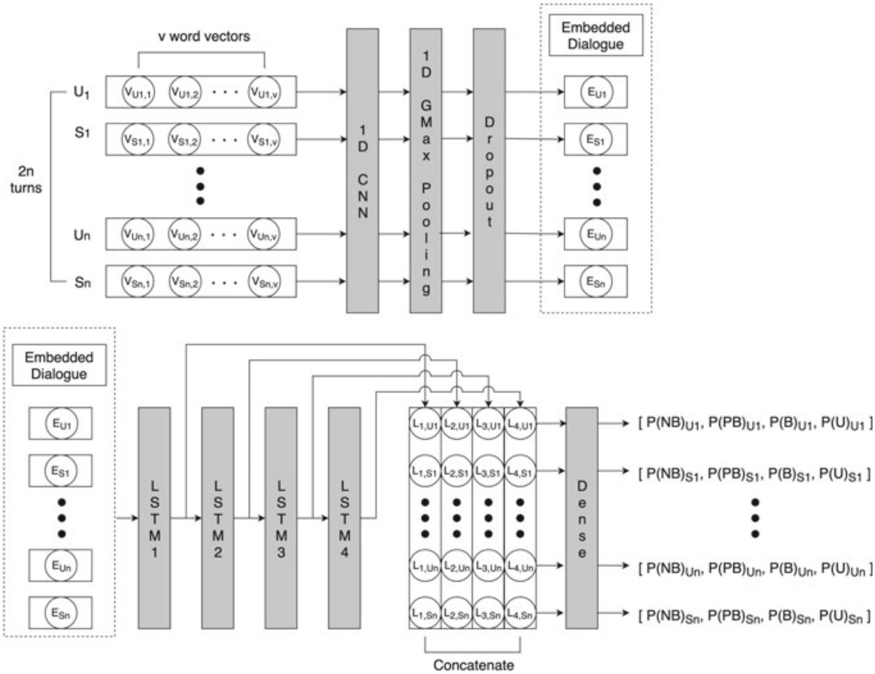


Fig. 1 Architecture diagram of our LSTM-based model

For the preprocessing of both English and Japanese data, we first follow the same approach as RSL17BD at DBDC3 to produce a sequence of 300 dimensional word embedding vectors for each utterance in every dialogue. The number of word vectors in each sequence is fixed to v, with v set to 50. This is done by truncating sequences that are longer than v and padding sequences that are shorter than v with zero vectors. The number of turns in a each dialogue is fixed to $2n$ by either removing the first system utterance which has no annotations or removing the last user turn. For the English data and the Japanese data from DCM, DIT, and IRS, the number of turns in each dialogue is fixed to 20 by setting n to 10. For the data of the five dialogue systems under `dbd_livecompe_eval`, the number of turns in each dialogue is fixed to 30 by setting n to 15.

The process above produces a dialogue of $2n$ turns, with each turn represented by a sequence of v word vectors. We apply One-dimensional Convolutional Neural Networks (1D CNN), One-dimensional Global Max Pooling (1D GMax Pooling), and Dropout [11] for each sequence to produce an embedded dialogue. The 1D CNN layer uses 150 filters of size 2 with ReLU [9] as the activation function. The dropout rate of the Dropout layer is set to 0.4.

The embedded dialogue is then processed by 4 LSTM layers sequentially. Each LSTM layer contains 64 units, with dropout set to 0.1, and recurrent dropout set to 0.1. We used LSTM instead of Bi-LSTM because the usage of turns after the target

Table 1 Results of LSTM-based model with different configurations on DBDC3 English evaluation data

Model	Epochs	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
LSTM-ADAD-CAT	100	0.4130	**0.4616**	0.0928	0.0573
LSTM-ADAM-MSE	100	0.3940	0.3714	0.0516	0.0300
LSTM-CNN-ADAM-MSE	50	**0.4620**	0.4268	**0.0474**	**0.0274**

Table 2 Results of LSTM-based model with different configurations on DBDC3 Japanese evaluation data

Model	Epochs	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
LSTM-ADAM-MSE	100	0.5448	0.6148	0.0885	0.0497
LSTM-CNN-ADAM-MSE	50	**0.5739**	**0.6594**	**0.0826**	**0.0463**

system utterance is disallowed. The output sequences from the 4 LSTM layers are concatenated to form a $(2n, 256)$ dimensional matrix, and processed by a Dense layer with softmax activation and 4 outputs. The 4 outputs represent P(NB), P(PB), P(B), and P(U) respectively. The probability distribution for each target system utterance is calculated by normalising P(NB), P(PB), and P(B) by their sum. The single breakdown label is determined by choosing the label with the highest probability in the distribution.

The modifications above are decided by training and evaluating different model configurations using the English and Japanese data from DBDC3. The evaluation results are shown in Tables 1 and 2. **LSTM** means the model utilises LSTM, and **LTSM-CNN** means the model utilises LSTM and CNN. **ADAD-CAT** means the model utilises Adadelta as optimizer and categorical cross entropy as loss function, and **ADAM-MSE** means the model utilises Adam as optimizer and mean squared error as loss function.

The results show that for English data, **LSTM-CNN-ADAM-MSE** outperformed **LSTM-ADAD-CAT** and **LSTM-ADAM-MSE** in all evaluation metrics except F1 (B). Although **LSTM-ADAD-CAT** achieved high performance in F1 (B), its performance in mean squared error (MSE (NB, PB, B)) was poor. Since mean squared error is emphasised in this challenge, we decided to discard **LSTM-ADAD-CAT**. For Japanese data, **LSTM-CNN-ADAM-MSE** outperformed **LSTM-ADAM-MSE** in all evaluation metrics. We did not evaluate **LSTM-ADAD-CAT** because it is already discarded after the evaluation of English data. In the end, we chose to utilise the configuration of **LSTM-CNN-ADAM-MSE** when submitting the models for Run 2 and Run 3.

4 Runs

Our five runs are described in Sects. 4.1–4.5. In Runs 1–3, we used the same strategy in creating the training data from the given development data in DBDC4. For the English submission, we created a single group of training data which consists of the entire English development data. We refer it as E_{t1} hereinafter. The entire English evaluation data is referred as E_{e1} hereinafter. For the Japanese submission, we created two groups of training data. The first group consists of the development data from DCM, DIT, and IRS, and the second group consists of the development data from the five dialogue systems under `dbd_livecompe_eval`. We refer them as J_{t1} and J_{t2} hereinafter. The evaluation data from DCM, DIT, and IRS and the evaluation data from the five dialogue systems under `dbd_livecompe_eval` are referred as J_{e1} and J_{e2} hereinafter.

4.1 Run 1: Decision Tree-Based Model

For the English submission, we trained our Decision Tree-based model with E_{t1} and made predictions on E_{e1}. For the Japanese submission, we built two models by training one with J_{t1}, and the other with J_{t2}. We made predictions on J_{e1} using the former model and J_{e2} using the latter model.

4.2 Run 2: LSTM-Based Model

For the English submission, we pretrained our LSTM-based model for 30 epochs with the entire English development and evaluation data in DBDC3, fine-tuned it by training for 32 epochs with E_{t1}, and made predictions on E_{e1}. For the Japanese submission, we built two LSTM-based models. The first model is trained for 30 epochs with J_{t1}. The second model is created by loading the weights from the first model and fine-tuning for 25 epochs with J_{t2}. We made predictions on J_{e1} using the first model and J_{e2} using the second model. Every model is trained using a batch size of 32.

4.3 Run 3: Ensemble of 5 LSTM-Based Models

The way an ensemble of 5 LSTM-based models is built is described as follows: Given training data D_t and evaluation data D_e, we randomly divide D_t into 10 portions and sample 5 portions from it. We build 5 models, where each model is trained using one of the sampled portions as validation data and the rest of the development data

Table 3 Results of each LSTM-based model in Run 3 on the sampled validation data from E_{t1}

Model	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
1	0.5286	0.5385	0.0649	0.0343
2	0.5190	0.5318	0.0701	0.0370
3	0.5524	0.5660	0.0713	0.0375
4	0.5381	0.6306	0.0706	0.0370
5	0.5810	0.6635	0.0771	0.0401

Table 4 Results of each LSTM-based model in Run 3 on the sampled validation data from J_{t1}

Model	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
1	0.5664	0.5782	0.0887	0.0469
2	0.5804	0.6057	0.0914	0.0477
3	0.5944	0.6505	0.0786	0.0429
4	0.5944	0.6402	0.0903	0.0473
5	0.5846	0.6231	0.0961	0.0495

as training data. The batch size is set to 32. Each model is saved when the validation loss is minimum and no overfitting occurred. We make predictions on D_e using each model, and take the mean of the predicted probability distribution for each target system utterance from the 5 models to produce a new probability distribution. The new single breakdown label is determined by choosing the label with the highest probability in the new probability distribution.

For the English submission, we pretrained an LSTM-based model for 30 epochs with the entire English development and evaluation data in DBDC3. The ensemble of 5 LSTM-based models is built by fine-tuning the pretrained model with $D_t = E_{t1}$ and $D_e = E_{e1}$. The results of each LSTM-based model on the sampled validation data are shown in Table 3.

For the Japanese submission, we built two ensemble models. The first model is built with $D_t = J_{t1}$ and $D_e = J_{e1}$. The results of each LSTM-based model on the sampled validation data are shown in Table 4. The second model is built by loading the weights of the first model from the Japanese submission in Run 2 and fine-tuning it with $D_t = J_{t2}$ and $D_e = J_{e2}$. The results of each LSTM-based model on the sampled validation data are shown in Table 5.

4.4 Run 4: Ensemble of Run 1 and Run 2

For both English and Japanese submissions, we take the mean of the predicted probability distribution for each target system utterance from Run 1 and Run 2 to produce a new probability distribution. The new single breakdown label is determined by choosing the label with the highest probability in the new probability distribution.

Table 5 Results of each LSTM-based model in Run 3 on the sampled validation data from J_{t2}

Model	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
1	0.6313	0.4583	0.0706	0.0371
2	0.6953	0.4324	0.0591	0.0323
3	0.6484	0.3750	0.0510	0.0277
4	0.6797	0.3529	0.0600	0.0319
5	0.6016	0.0000	0.0678	0.0336

Table 6 Official results on English data

Model	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
Run 1	0.4990	0.4411	0.0700	0.0362
Run 2	0.4730	0.4483	0.0725	0.0374
Run 3	0.5200	0.4554	0.0675	0.0346
Run 4	0.5050	0.4650	0.0690	0.0353
Run 5	**0.5255**	**0.4690**	**0.0662**	**0.0336**

4.5 Run 5: Ensemble of Run 1 and Run 3

This run is identical with Run 4 except that Run 2 is replaced by Run 3.

5 Results

Tables 6 and 7 show the official results of our English and Japanese runs respectively. It can be observed that Run 5 did well on average. For English runs, it outperformed all other runs in all evaluation metrics. For Japanese runs, it outperformed all other runs in JSD (NB, PB, B) and MSE (NB, PB, B).

Tables 8 and 9 show the results of comparing the MSE (NB, PB, B) of Runs 1–5 based on the Randomised Tukey's Honestly Significant Differences (HSD) test. The test is conducted with 10,000 replicates. The p-values are shown alongside with effect sizes (standardised mean differences) [10]. Table 8 shows that Run 5 statistically significantly outperformed all other runs in terms of MSE (NB, PB, B) for the English data. Table 9 shows that Run 5 statistically significantly outperformed all other runs except Run 4 in terms of MSE (NB, PB, B) for the Japanese data. The p-values show that the differences are statistically significant at the alpha level of 0.05.

Table 7 Official results on Japanese data

Run	Accuracy	F1 (B)	JSD (NB, PB, B)	MSE (NB, PB, B)
Run 1	0.5390	0.4568	0.0975	0.0492
Run 2	0.5412	**0.4613**	0.0989	0.0509
Run 3	**0.5476**	0.4589	0.0967	0.0493
Run 4	0.5412	0.4583	0.0954	0.0480
Run 5	0.5444	0.4603	**0.0947**	**0.0475**

Table 8 P-value based on randomised Tukey's HSD test/effect sizes for MSE (NB, PB, B) (English)

	Run2	Run3	Run4	Run 5
Run 1	$p = 0.007(-0.110)$	$p < 0.0001(0.139)$	$p = 0.0669(0.080)$	$p < 0.0001(0.227)$
Run 2	–	$p < 0.0001(0.249)$	$p < 0.0001(0.191)$	$p < 0.0001(0.337)$
Run 3	–	–	$p = 0.387(-0.059)$	$p = 0.0415(0.088)$
Run 4	–	–	–	$p < 0.0001(0.146)$

Table 9 P-value based on randomised Tukey's HSD test/effect sizes for MSE (NB, PB, B) (Japanese)

	Run2	Run3	Run4	Run 5
Run 1	$p = 0.0086(-0.104)$	$p = 1(-0.002)$	$p = 0.0338(0.076)$	$p < 0.0001(0.112)$
Run 2	–	$p = 0.0086(0.102)$	$p < 0.0001(0.181)$	$p < 0.0001(0.216)$
Run 3	–	–	$p = 0.0222(0.079)$	$p < 0.0001(0.114)$
Run 4	–	–	–	$p = 0.6577(0.035)$

6 Discussions

6.1 Naive Strategy in Creating the Training Data

As described in Sect. 4, in Runs 1–3, we used the same strategy in creating the training data from the given development data. For the English submission, we created one group of training data E_{t1} and trained a single model with it. The reason for doing so is that we wanted to create sufficient training data, since there is only a total number of 211 dialogues. For the Japanese submission, we created two groups of training data J_{t1} and J_{t2} and trained two models with them respectively. The reason for doing so is that the first group consists of dialogues with 21 turns (fixed to 20 turns in preprocessing) while the second group consists of dialogues with 31 turns (fixed to 30 turns in preprocessing). Because our LSTM-based model only accepts fixed turn lengths, we had to build two models to target two different turn lengths. We used the

Table 10 Official results of MSE (NB, PB, B) for J_{e2}

	IRS	MMK	MRK	TRF	ZNK
Run 1	0.0662	0.0243	0.0393	0.0282	0.0418
Run 2	0.0606	0.0184	0.0328	0.0230	0.0389
Run 3	0.0602	0.0195	0.0322	0.0231	0.0394
Run 4	0.0606	0.0197	0.0341	0.0236	0.0378
Run 5	0.0608	0.0206	0.0340	0.0239	0.0384

same strategy for building our Decision Tree-based model so that the ensemble with the LSTM-based model can be done easily.

Nevertheless, the above strategy is rather naive as it does not consider the overall probability distribution of the three breakdown labels for each dialogue system. By analysing the dataset,[3] we found out that system IRS in J_{t2} has a significantly higher probability for label B compared to the other four systems. We believe that IRS should not have been combined with the other four systems to create training data J_{t2}. Furthermore, the model which is trained with J_{t2} should not have been used for predicting the data of IRS in J_{e2}.

Table 10 shows the official results of MSE (NB, PB, B) for J_{e2}. It can be observed that due to the naive strategy above, all runs achieved poor performance with regard to IRS. To improve the result, we believe that the development data of IRS should be excluded from J_{t2} and combined with J_{t1}. When predicting the labels for IRS in J_{e2}, we should utilise the model trained with J_{t1} instead of the one trained with J_{t2}. This proposed strategy requires us to either fix all training data to 30 turns in the LSTM-based model or develop a new model which accepts a shorter fixed turn length such as 5.

6.2 *Ensemble Works?*

We analysed our runs in terms of MSE (NB, PB, B) (referred as MSE in this section), which is the emphasised evaluation metric in this challenge. From Tables 6 and 7, it can be observed that Run 4 outperformed Run 1 and Run 2, and Run 5 outperformed Run 1 and Run 3 in terms of MSE for both English and Japanese data. To investigate how well the ensemble actually worked for each utterance, we would like to know the number of target system utterances for which the ensemble model outperformed the original models that were ensembled. In this section, we focus on Run 5 which

[3]The detailed analysis is shown in the full version of this paper [12].

Table 11 Number of turns for which each run outperformed the others for the English data

A subset of turns V' ($\subset V$)	$\|V'\|$
$V_{1<3,5}$	866
$V_{3<1,5}$	958
$V_{5<1,3}$	176
$\{v \| mse_1(v) < mse_5(v) \wedge mse_3(v) < mse_5(v), v \in V\}$	0

Table 12 Number of turns for which each run outperformed the others for the Japanese data

A subset of turns V' ($\subset V$)	$\|V'\|$
$V_{1<3,5}$	1200
$V_{3<1,5}$	1233
$V_{5<1,3}$	162
$\{v \| mse_1(v) < mse_5(v) \wedge mse_3(v) < mse_5(v), v \in V\}$	0

achieved the best performance in terms of MSE and compare its results with Run 1 and Run 3.[4]

Tables 11 and 12 show the number of target system utterances for which each run outperformed the others. V denotes the set of target system utterances in the evaluation dataset, and $mse_i(v)$ denotes the MSE of Run i given a target system utterance v ($\in V$). $V_{1<3,5}$, $V_{3<1,5}$, and $V_{5<1,3}$ are defined by the following equations:

$$V_{1<3,5} = \{v | mse_1(v) < mse_5(v) < mse_3(v), v \in V\}, \tag{1}$$

$$V_{3<1,5} = \{v | mse_3(v) < mse_5(v) < mse_1(v), v \in V\}, \tag{2}$$

$$V_{5<1,3} = \{v | mse_5(v) < mse_1(v) \wedge mse_5(v) < mse_3(v), v \in V\}, \tag{3}$$

From Tables 11 and 12, it can be observed that the number of target system utterances for which Run 5 outperformed the other runs is relatively small.

Tables 13 and 14 show the mean MSE of Run 1, Run 3, and Run 5 over $V_{1<3,5}$, $V_{3<1,5}$, and $V_{5<1,3}$ respectively. From Tables 13 and 14, it can be observed that when Run 5 outperformed Run 1 and Run 3, the MSEs of Run 1 and Run 3 tend to be low. We plotted the relationship of the MSE between Run 1 and Run 3 and discovered that Run 5 tends to outperform the two other runs when the MSE of Run 1 and Run 3 are similar.[5]

We looked into the system utterances for which the difference between the MSE of Run 1 and Run 3 are high and found out these utterances tend to be labeled with high

[4]When comparing the runs in Sect. 6.2, we remove the first predicted system utterance of every dialogue in Japanese data. This is because the first system utterances in Japanese data are all annotated with the same labels (NB) and are all predicted correctly with MSE = 0.0 by every run.

[5]All of the plotted figures and detailed analysis are shown in the full version of this paper [12].

Table 13 Mean MSE over $V_{1<3,5}$, $V_{3<1,5}$ and $V_{5<1,3}$ for the English data

A subset of turns V' ($\subset V$)	Run 1	Run 3	Run 5
$V_{1<3,5}$	0.0270	0.0451	0.0344
$V_{3<1,5}$	0.0481	0.0285	0.0367
$V_{5<1,3}$	0.0159	0.0159	0.0129

Table 14 Mean MSE over $V_{1<3,5}$, $V_{3<1,5}$ and $V_{5<1,3}$ for the Japanese data

A subset of turns V' ($\subset V$)	Run 1	Run 3	Run 5
$V_{1<3,5}$	0.0463	0.0721	0.0573
$V_{3<1,5}$	0.0649	0.0399	0.0505
$V_{5<1,3}$	0.0195	0.0194	0.0164

probability of NB or B compared to other utterances. We plotted the relationship of the absolute difference between the MSE of Run 1 and Run 3 and $\max\{p^*(NB), p^*(B)\}$, where $\max\{p^*(NB), p^*(B)\}$ denotes the maximum probability of the labeled probabilities of NB and B. Through observing the figure, we found out that the MSE of Run 1 and Run 3 tend to be similar when $\max\{p^*(NB), p^*(B)\}$ is low. This means that the ensemble model tends to perform the best in target system utterances which are not labeled with high probability of NB or B. Therefore, to further improve our ensemble model, we should either develop a new ensemble strategy different from simple averaging or include a third model which focuses on minimising the MSE in target system utterances that are labeled with high probability of NB or B.

7 Conclusions

We submitted five runs to both English and Japanese subtasks of DBDC4. Run 1 utilises a Decision Tree-based model; Run 2 utilises an LSTM-based model; Run 3 performs an ensemble of 5 LSTM-based models; Run 4 performs an ensemble of Run 1 and Run 2; Run 5 performs an ensemble of Run 1 and Run 3. Run 5 statistically significantly outperformed all other runs in terms of MSE (NB, PB, B) for the English data and all other runs except Run 4 in terms of MSE (NB, PB, B) for the Japanese data (alpha level = 0.05).

Our future work includes utilising a proposed strategy in creating the training data and improving our ensemble model. The proposed strategy considers the overall probability distribution of the three breakdown labels for each dialogue system and requires us to either fix all training data to 30 turns in the LSTM-based model or develop a new model which accepts a shorter fixed turn length such as 5. To improve our ensemble model, we should either develop a new ensemble strategy different from simple averaging or include a third model which focuses on minimising the MSE in target system utterances that are labeled with high probability of NB or B.

References

1. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
2. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42. https://doi.org/10.1007/s10994-006-6226-1
3. Higashinaka R, D'Haro LF, Shawar BA, Banchs R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, ao Sedoc J, Overview of the dialogue breakdown detection challenge 4
4. Higashinaka R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Kaji N (2017) Overview of dialogue breakdown detection challenge 3. In: Proceedings of dialog system technology challenge 6 (DSTC6) workshop
5. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
6. Kato S, Sakai T (2017) RSL17BD at DBDC3: computing utterance similarities based on term frequency and word embedding vectors. In: Proceedings of dialog system technology challenge 6 (DSTC6) workshop
7. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization
8. Lopes J (2017) How generic can dialogue breakdown detection be? The KTH entry to DBDC3. In: Proceedings of dialog system technology challenge 6 (DSTC6) workshop
9. Nair V, Hinton GE (2010) Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th international conference on machine learning, ICML'10, pp 807–814
10. Sakai T (2018) Laboratory experiments in information retrieval: sample sizes, effect sizes, and statistical power, vol 40(1). https://doi.org/10.1007/978-981-13-1199-4, https://www.springer.com/jp/book/9789811311987
11. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
12. Wang CH, Kato S, Sakai T (2019) RSL19BD at DBDC4: ensemble of decision tree-based and LSTM-based models. arXiv:1905.01799
13. Zeiler MD (2012) ADADELTA: an adaptive learning rate method. CoRR. arXiv:1212.5701

LSTM for Dialogue Breakdown Detection: Exploration of Different Model Types and Word Embeddings

Mariya Hendriksen, Artuur Leeuwenberg, and Marie-Francine Moens

Abstract One of the principal problems of human-computer interaction is miscommunication. Occurring mainly on behalf of the dialogue system, miscommunication can lead to dialogue breakdown, i.e., a point when the dialogue cannot be continued. Detecting breakdown can facilitate its prevention or recovery after breakdown occurred. In the paper, we propose a multinomial sequence classifier for dialogue breakdown detection. We explore several LSTM models each different in terms of model type and word embedding models they use. We select our best performing model and compare it with the performance of the best model and with the majority baseline from the previous challenge. We conclude that our detector outperforms the baselines during the offline testing.

1 Introduction

The importance of spoken dialogue systems has been steadily increasing over the years. Some of the reasons for such a popularity raise include their ability to provide instant twenty-four-hour service. and applicability across different domains such as website assistance, education, customer service, e-commerce, and entertainment.

Naturally, the usefulness of a dialogue system largely depends on its ability to interact with users. One of the major obstacles on the way to the goal is miscommu-

Work performed while at KU Leuven.

M. Hendriksen (✉)
University of Amsterdam, Amsterdam, The Netherlands
e-mail: m.hendriksen@uva.nl

A. Leeuwenberg · M.-F. Moens
KU Leuven, Heverlee, Belgium
e-mail: tuur.leeuwenberg@kuleuven.be

M.-F. Moens
e-mail: sien.moens@kuleuven.be

E. Marchi et al. (eds.), *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, Lecture Notes in Electrical Engineering 714,
https://doi.org/10.1007/978-981-15-9323-9_41

nication. We define miscommunication as a situation when a dialogue system gives a user an inappropriate reply. In other words, we assume that miscommunication occurs on the behalf of a system.

Miscommunication can lead to dialogue breakdown, i.e., a point in dialogue when the interaction is interrupted with or without completion of the performed task [13]. In this perspective, a model capable of detecting dialogue breakdown points can enhance the quality of human-computer interaction. For instance, such a detector could help to avoid system responses which cause a breakdown or identify breakdowns after they occur and launch the procedures necessary to get out of the breakdown situation.

The rest of the paper is organized as follows: Sect. 2 describes the task of dialogue breakdown detection and the dataset which was used for model training. In the Sect. 3, we discuss the related work on dialogue breakdown detection models, applications of dialogue breakdown detection detectors, the taxonomy of errors causing dialogue breakdown and the alternative methods for system response assessment. We describe the proposed model in the Sect. 4 and explain the experiment setting in Sect. 5. The discussion of the results is presented in Sect. 6. In Sect. 7, we draw conclusions and suggest directions for future work.

2 Task Description

We address the task of dialogue breakdown detection. The task was introduced as a challenge in [17]. Since the introductions, three challenges were held [5, 6]. We submit the model for the fourth challenge [7].

In particular, we aim to develop a system to predict whether a given system utterance causes a breakdown. The prediction is to be based both on the current utterance and on the dialogue history. Each system response is to be marked with one of the three labels:

- NB—not a breakdown: it is possible to continue the dialogue smoothly.
- PB—possible breakdown: it is difficult to continue the dialogue smoothly.
- B—breakdown: it is difficult to continue the dialogue.

For model development, we utilize the training dataset provided by the challenge organizers. The set consists out of 211 dialogues. Each dialogue in the set has a length of 20 or 21 utterances: 25 dialogues have length 20, whereas 186 dialogues have length 21.

Every system response is labelled by fifteen annotators. Hence, for every system utterance, the model predicts the probability distribution of labels and assign the label with the highest probability to given system response.

There are two primary types of metrics for performance evaluation: distribution-related and classification-related. Following the findings presented in [20], we focus on mean squared error MSE(NB,PB,B) as the primary metric.

3 Literature Review

In this paper, we describe the model for dialogue breakdown detection. Related work falls in the following areas: (1) existing dialogue breakdown detection models, (2) applications for dialogue breakdown detection systems, (3) analysis of errors causing dialogue breakdown, and (4) the alternative techniques for dialogue system response assessment.

There are exist several models for dialogue breakdown detection. The list includes Conditional Random Fields model which is used as a baseline in the challenge [7], Extremely Randomized Trees [11], Maximum Entropy model [6], Support Vector Machines [12], Memory Networks [9] and Recurrent Neural Networks (RNN) [14], in particular long short-term memory networks (LSTM) [12, 21]. Besides, some systems feature attention modules as part of their architecture [9, 14].

Dialogue breakdown detection can be used to re-rank responses of a chat-oriented dialogue system. In [10], the authors suggest three re-ranking approaches: classification, regression, and probability-based approach. The classification technique was based on the classification of all the possible system responses. The regression method implied the application of linear regression with the probability distribution of breakdown labels and response scores as a feature set. In the case of the probability-based approach, the non-breakdown probability was used for re-ranking.

Other application examples include [18] where the author applies dialogue breakdown detection system for selecting tweets that can be used as responses of a chat-oriented dialogue system.

Another direction of research in the field is to investigate the errors causing a breakdown in chat-oriented dialogue systems. This is done in [4] where researchers present a taxonomy of this type of errors. Inspired by the Gricean maxims,[1] the authors define utterance-level, environmental-level, and cooperativeness error. The breakdown detector based on the taxonomy of errors is presented in [8].

Breakdown detection is not the only way to evaluate chatbot responses. For instance, [22] offer a similar technique for assessment of chatbot responses in non-task-oriented dialogues. In particular, they suggest measuring the appropriateness of utterances and customer satisfaction.

4 Proposed Model

In this section, we describe the proposed dialogue breakdown detector as well as the motivation behind the selected architecture and components (Fig. 1).

[1]The principles of effective communication in standard social setting [3].

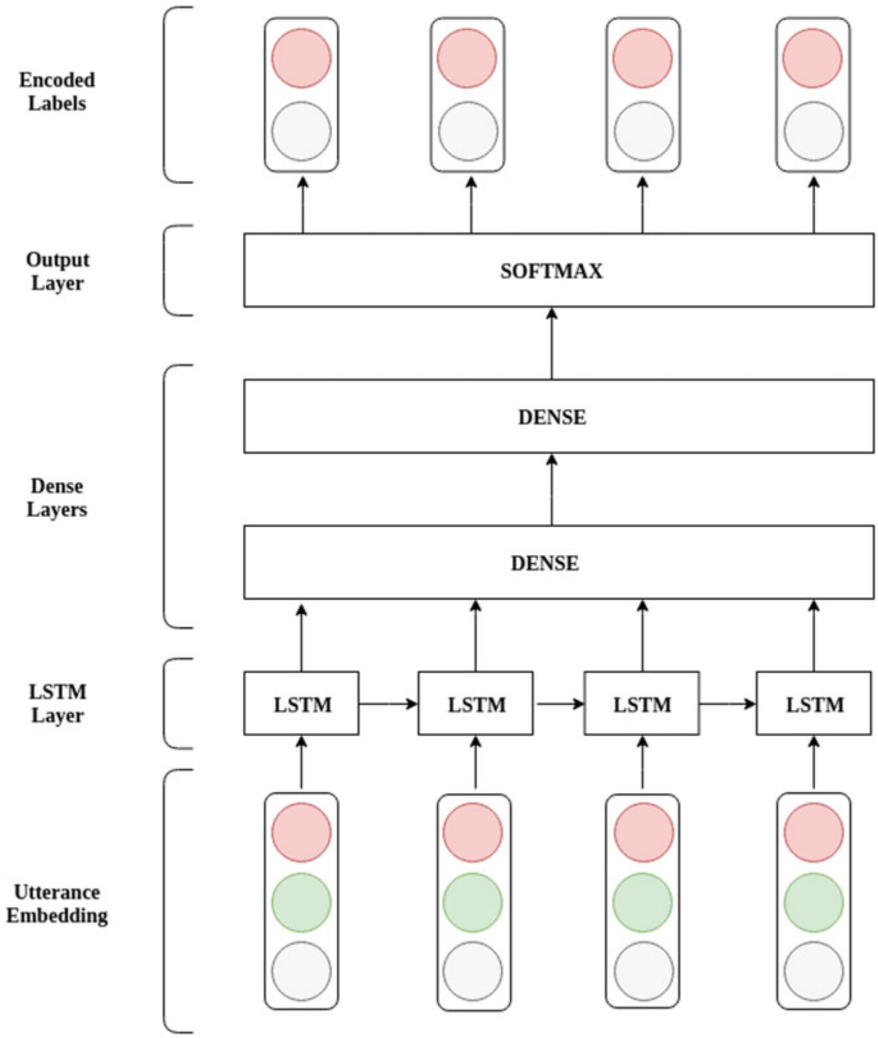


Fig. 1 Proposed model architecture: input layer, LSTM layer, two dense layers and a softmax output layer. The model takes dialogue representation as an input and assigns a label for every utterance

4.1 Model Type

After considering the existing dialogue breakdown detectors and their performances, we chose LSTM for their ability to process sequential data and handle long-term dependencies. Several LSTM models are considered: vanilla LSTM, stacked LSTM, and bidirectional LSTM.

4.2 Word Embedding Model

Another important aspect to consider when working with textual data is text representation. For this reason, we considered several word embedding types:

word2vec Google News
The first type of the word representations we use is the word2vec vectors pretrained on Google News corpora.[2] We use those vectors because they were featured in the LSTM model which demonstrated the best performance in terms of accuracy during DBDC3 [12]. The vectors were produced by a bag-of-words model (BoW) trained with negative sampling with window size 5. Each word is represented with an embedding of size 300.

GloVe Twitter
The second type of word representation we use is GloVe trained on Twitter data.[3] We use this embedding type because of the proximity of the Twitter domain to the task domain [14]. The vectors were obtained by training on 2 billions of tweets with representations for 27 billion tokens and a vocabulary of 1,2 million. The vectors are presented in 25d, 50d, 100d and 200d, we decided to use 200d vectors. The words are uncased.

GloVe Common Crawl
The third type of word embedding model is GloVe Common Crawl.[4] It contains representations for 840 billion tokens with the vocabulary of 2.2 million. The words are cased and the dimensionality of the vectors is 300d. We use this embedding type because, unlike the GloVe Twitter, it is domain-independent.

4.3 Model Architecture

The proposed model takes dialogue representations as input. Dialogue representations are composed out of utterance representations which, in turn, are created out of word representations. During the training phase, the training loss is computed and models parameters are optimized.

---

[2]Google News 100B 3M words, URL: https://github.com/3Top/word2vec-api, last checked on 07-04-2019.

[3]GloVe: Twitter, URL: https://nlp.stanford.edu/data/glove.twitter.27B.zip, last checked on 27-03-2019.

[4]GloVe: Common Crawl, URL: https://nlp.stanford.edu/data/glove.840B.300d.zip, last checked on 27-03-2019.

4.3.1 Input Representation

Word representations are acquired with pretrained word embedding vectors. We represent all the out of vocabulary (OOV) tokens with the token unk. Each sentence embedding is represented as the average of the token embeddings that comprise the sentence. Consequently, each utterance embedding has the same dimensionality as the word embedding. Dialogue is represented as a sequence of user and system utterances. We pad dialogues to ensure that each dialogue has a length of 21 utterances.

4.3.2 Target Representation

Each system response in a dialogue is labelled with B (breakdown), PB (possible breakdown) or NB (not a breakdown) label. Besides, we introduce label U to mark user utterance. The labels are represented with one-hot encoding. Therefore, each sequence of dialogue labels is represented as 21×4 matrix.

4.3.3 Loss

In the given task, for each utterance in the dialogue, the network predicts a probability for each of the labels and compares it with the ground truth. Hence, the model should use a loss function that would compare the labels probability distribution with the ground truth and penalize incorrect label prediction. One of the suitable objective function for this task is cross-entropy function [1]. The function measures the difference between two probability distributions. The function is computed as follows:

$$H(y, \hat{y}) = -\sum_{i=1} y_i log(\hat{y}_i) \tag{1}$$

where y is the ground truth label and $\hat{y}$ is the predicted label.

4.3.4 Optimization

We optimize model with mini-batch gradient descent of a batch size of 4. We use Root Mean Square Propagation (RMSProp) [19] which is the extension of Resilient Backpropagation (Rprop) learning [16]. RMSProp combines the robustness of Rprop, the efficiency of mini-batches and the effective averaging of gradients over mini-batches.

5 Experiment

In this section, we discuss dataset preprocessing, define the baselines, explain the training pipeline and present the results.

5.1 Data Preprocessing

Before starting preprocessing, we have to make several decisions:

- Do we keep user turns? Following the findings described in [12], we decided to keep the user turns as they enhance detector performance.
- How do we feed user turns to the model? Initially, we considered concatenating user turns with the corresponding system response into exchange pairs. Such an approach would allow avoiding the introduction of an extra label. However, results shown in [21] testify that such concatenation decreases model performance. Hence, we resolve to mark each user utterance with an extra label *U* and feed them to the models as a separate utterance.

After making the decisions, we can start dataset preprocessing. Since we use three different types of embedding models, we prepare one dataset for each of the types. We do it in order to take all the particularities of the embedding models into account. Two major preprocessing steps were applied to all three datasets: tokenization and replacement of apostrophe contractions.

In general, all the datasets are tokenized with `TweetTokenizer`.[5] This tokenizer is a part of `casual` submodule of `nltk.tokenize` package, it was selected because the domain of its primary use is closely related to the domain of the task. In particular, `TweetTokenize` is able to handle emoticons the dataset contain.

In addition to tokenization, extra rules are applied to common apostrophes contractions. For example, the contraction $that's$ is transformed to *that is*.

Besides the mentioned preprocessing steps, we remove punctuation signs from the dataset for word2vec Google News vectors because the model did not know any punctuation signs. Additionally, we lowercase all the words in the dataset for GloVe Twitter because the pretrained word vectors are uncased.

As mentioned in the Sect. 2, dialogue length varies from 20 to 21 utterances per dialogue. In such a case, we can either truncate or pad the dialogues. Since the first approach implies a loss of certain parts of data, we implement padding as a more suitable option.

[5]NLTK 3.4 documentation, URL: http://www.nltk.org/api/nltk.tokenize.html, last checked on 27-03-2019.

5.2 Model Training

The model hyperparameters were determined by a grid search. The model was trained for a maximum of 100 epochs. To prevent overfitting, we employ two regularization techniques: early stopping and dropout (both standard dropout and recurrent dropout).

5.3 Baselines

We compare the performance of our model with the two baselines defined during DBDC3. First is the majority baseline, the second model is the best performing model of DBDC3 which was an attention-based detector [14].

5.4 Model Selection

The next step is to select the best performing model out of the nine models we experiment with.

In order to investigate which type of LSTM produces the best performance, we compare the metrics results. We do this by calculating the average metric score for each $Model \times Metric$ pair across three embedding types. The results are presented in the Table 1. Overall, it can be concluded that, given that all the metrics have equal importance, the best performance is obtained by vanilla LSTM, stacked LSTM is the second best, and Bi-LSTM is the worst.

Next, we turn to the investigation of the relationship between model performance and its embedding type. In analogy with the above-mentioned idea, we calculate an average performance score for each metric. The results presented in the Table 2, allow to conclude that GloVe Common Crawl demonstrate the best performance, the GloVe Twitter being the second best, the word2vec Google News is the worst.

Table 1 Average metric scores for every LSTM type

LSTM type	MSE(NB,PB,B)
Vanilla LSTM	0.0213
Stacked LSTM	0.0224
Bi-LSTM	0.0222

Table 2 Average metric scores for every word embedding type

Embedding type	MSE(NB,PB,B)
word2vec Google News	0.0351
GloVe Twitter	0.0301
GloVe Common Crawl	0.0213

6 Discussion and Implications

After running experiments with the created models, we concluded that the vanilla LSTM with GloVe Common Crawl embedding demonstrates the best performance. For this reason, we compare it with the selected baselines. The comparison is given in the Table 3. As can be seen, our model outperforms the baselines.

6.1 Analysis of Error Patterns

Comprehension of patterns in the type of errors that the best models make could provide additional insight into their improvement.

During the experiments, we found out that embedding type impacts model performance. In particular, we discovered that there is a negative correlation between the proportion of OOV and model performance. Hence, we investigate the type of OOV tokens. In general, GloVe pretrained on Common Crawl does not know 245 tokens in the dataset (or 115 unique tokens).

Emoticons

One of the significant parts of OOV tokens were emoticons. The emoticons are the strong cues of how the user feels and hence can help to understand when the conversation goes in the wrong direction. Therefore, they should be taken into account. The model's vocabulary includes basic emoticons such as ☺ or ☹ but not more complex ones such as 😵 or 😎. Besides, some dialogues featured emoticons represented as capitalized descriptions in square brackets (e.g., [SMILING FACE WITH SUNGLASSES], [SPEAK-NO-EVIL MONKEY]). As a possible solution to the problem,

Table 3 Comparison of the created model with the baseline models

Model	MSE(NB,PB,B)
Majority baseline	0.0224
NCDS	0.0237
Proposed model	0.0213

it might be helpful to replace the complex emoticons with their basic counterparts to facilitate the model's understanding by reducing variance.

Apostrophe Contractions
Many of the tokens were not recognized because they represented apostrophe contractions. The examples included both straightforward cases like *isn't* (*is not*) and ambiguous situations such as *he's* which depending on context can be interpreted either like *he has* or *he is*. In general, the problem can be resolved by the introduction of extra contraction replacement rules. In the case of ambiguous situations, it might be better to apply a disambiguation procedure first.

Abbreviations
Another significant group of unknown tokens includes abbreviations, e.g., *ConvAI (conversational Artificial Intelligence)*. The issue can be addressed by abbreviation expansion. For instance, we could create a dictionary containing the most common abbreviations.

Misspellings
Some other words are not recognized because of misspelling. The most common type of mistakes is skipping white-space and thereby merging words. E.g., 'questionWho' and 'wait.Where'. Splitting these words while preprocessing the datasets can help to address the issue. Moreover, there were standard spelling mistakes such as 'seee' or 'tommorow'. Such standard mistakes can be resolved by adding a spellchecker.

7 Conclusion and Future Work

In this paper, we present a model for dialogue breakdown detector. Offline testing demonstrates that our model outperforms the best-performing model from DBDC3. Additionally, we investigate how model architecture and word embedding model influence detector performance.

Future work will focus on further exploration of model architectures and word embeddings. In particular, it would be interesting to explore different types of attention mechanisms and investigate such embedding models such as BERT [2] and EMLo [15]. Another direction of future work includes dataset expansion, specifically in terms of different languages and modalities.

References

1. De Boer PT, Kroese DP, Mannor S, Rubinstein RY (2005) A tutorial on the cross-entropy method. Ann Oper Res 134(1):19–67
2. Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805
3. Grice HP (1975) Logic and conversation, pp 41–58

4. Higashinaka R, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y, Mizukami M (2015) Towards taxonomy of errors in chat-oriented dialogue systems. In: Proceedings of the 16th annual meeting of the special interest group on discourse and dialogue, pp 87–95
5. Higashinaka R, Funakoshi K, Inaba M, Arase Y, Tsunomori Y (2016) The dialogue breakdown detection challenge 2. In: Proceedings of SIG-SLUD
6. Higashinaka R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Kaji N (2017) Overview of dialogue breakdown detection challenge 3. In: Proceedings of dialog system technology challenge, vol 6
7. Higashinaka R, D'Haro LF, Shawar BA, Banchs R, Funakoshi K, Inaba M, Tsunomori Y, Takahashi T, Sedoc J (2019) Overview of the dialogue breakdown detection challenge 4. In: Proceedings of WOCHAT
8. Horii T, Araki M (2015) A breakdown detection method based on taxonomy of errors in chat-oriented dialogue (in Japanese)
9. Iki T, Saito A (2017) End-to-end character-level dialogue breakdown detection with external memory models
10. Inaba M, Takahashi K (2018) Improving the performance of chat-oriented dialogue systems via dialogue breakdown detection
11. Kato S, Sakai T (2017) RSL17BD at DBDC3: Computing similarities based on term frequency and word embedding vectors. In: Proceedings of DSTC6
12. Lopes J (2017) How generic can dialogue breakdown detection be? The KTH entry to DBDC3
13. Martinovsky B, Traum D (2006) The error is the clue: breakdown in human-machine interaction. Technical report, Institute for Creative Technologies, University of Southern California
14. Park C, Kim K, Kim S (2017) Attention-based dialog embedding for dialog breakdown detection. In: Dialog system technology challenges workshop (DSTC6)
15. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Proceedings of NAACL
16. Riedmiller M, Braun H (1993) A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE international conference on neural networks, 1993. IEEE, pp 586–591
17. Ryuichiro H, Funakoshi K, Kobayashi Y, Michimasa I (2016) The dialogue breakdown detection challenge: task description, datasets, and evaluation metrics
18. Sugiyama H (2016) Utterance selection based on sentence similarities and dialogue breakdown detection on NTCIR-12 STC task. In: Proceedings of NTCIR
19. Tieleman T, Hinton G (2012) Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: neural networks for machine learning. https://www.youtube.com/watch?v=ZgLyc7CYlWw
20. Tsunomori Y, Higashinaka R, Takahashi T, Inaba M (2018) Evaluating dialogue breakdown detection in chat-oriented dialogue systems. In: SEMDIAL
21. Xie Z, Ling G (2017) Dialogue breakdown detection using hierarchical bi-directional LSTMs. In: Proceedings of the dialog system technology challenges workshop (DSTC6)
22. Zeng Z, Luo C, Shang L, Li H, Sakai T (2017) Test collections and measures for evaluating customer-helpdesk dialogues. In: Proceedings of EVIA