SN Computer Science (2020) 1:246
https://doi.org/10.1007/s42979-020-00255-3

REVIEW ARTICLE

Review of State-of-the-Art Design Techniques for Chatbots

Ritu Agarwal[1] · **Mani Wadhwa**[1]

Received: 7 April 2020 / Accepted: 15 July 2020 / Published online: 29 July 2020

Abstract
Amazon's Alexa, Apple's Siri, Google Assistant and Microsoft's Cortana, clearly illustrate the impressive research work and potentials to be explored in the field of conversational agents. Conversational agent, chatter-bot or chatbot is a program expected to converse with near-human intelligence. Chatbots are designed to be used either as task-oriented ones or simply open-ended dialogue generator. Many approaches have been proposed in this field which ranges from earlier versions of hard-coded response generator to the advanced development techniques in Artificial Intelligence. In a broader sense, these can be categorized as rule-based and neural network based. While rule-based relies on predefined templates and responses, a neural network based relies on deep learning models. Rule-based are preferable for simpler task-oriented conversations. Open-domain conversational modeling is a more challenging area and uses mostly neural network-based approaches. This paper begins with an introduction of chatbots, followed by in-depth discussion on various classical or rule-based and neural-network-based approaches. The evaluation metrics employed for chatbots are mentioned. The paper concludes with a table consisting of recent research done in the field. It covers all the latest and significant publications in the field, the evaluation metrics employed, the corpus which is used as well as the possible areas of enhancement that exist in the proposed techniques.

Keywords AIML · Recurrent Neural Network · LSTM · Deep seq2seq · HRED

Introduction

In 1950, Alan Turing posed a question, Can machines think? [1] From that time onwards, a challenge has been posed to Artificial Intelligence practitioners to make machines think or in simple words disguise it as a human. Chatbots came into the picture as a utility program, an advisor or simply a friend with whom you can talk to. There are various design techniques which emerged during its evolution. This paper deals particularly with the techniques used to build chatbots and their respective chatbot example.

The primary task of a chatbot is to produce a suitable response by contemplating natural language input provided by humans. There are several ways to generate that response, which defines the modeling mechanism of a chatbot as shown in Fig. 1. One is the rule-based method, wherein clever parsing of user input with hardcoded phrases and pre-made templates are used to generate the reply. The other one, neural-network-based approach was made possible by the rise of deep learning. The neural network is trained on large data-sets so that it can generate relevant and grammatically correct responses. Input can be of any form-text, images or speech. So, the models have also been introduced to convert speech to text [2] and Convolutional Neural Network(CNN) [3] models enable chatbot to derive useful information from images [4].

The neural network-based approach can be broadly classified as retrieval based and generative. Retrieval based methods generate reply by computing the most relevant response, either based on the method of scoring function such as computing conditional probabilities [5] implemented through the neural network or by evaluating the relationship between context and candidate replies in a reinforced co-ranking manner [6]. On the contrary, Generative method produces one word at a time corresponding to the given input after probabilities have been computed over the whole vocabulary [7]. Procedure to combine both retrieval and generative

This article is part of the topical collection "Advances in Computational Approaches for Artificial Intelligence, Image Processing, IoT and Cloud Applications" guest edited by Bhanu Prakash K N and M. Shivakumar".

✉ Mani Wadhwa
mani.wadhwa100@gmail.com

[1] Department of Information Technology, Delhi Technological University, Delhi 110042, India

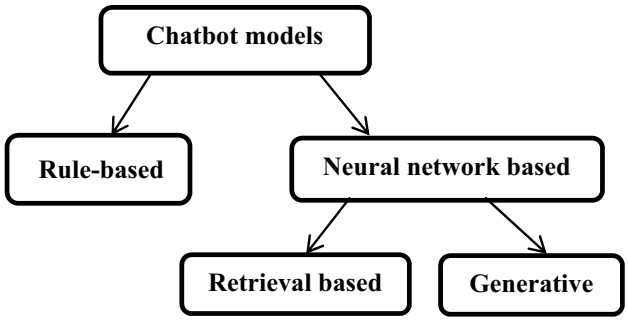

Fig.1 Classification of chatbot models

has also been introduced, wherein the retrieved reply is fed to generative model and the final response is decided by making a comparison between retrieved and generated reply based on reranking [8]

In terms of functionality, chatbots are mainly of two types. One is task-oriented, which are not the best conversational agent, but are very robust when it comes to executing specific tasks and handling domain-specific orders. Their application varies from making a restaurant reservation, booking flight tickets, promoting movies etc. The second type is open-domain chatbots. These are the typical conversational agents that try to mimic humans. Their aim is to generate human-like responses and get the other person into believing that it indeed is one. Every year, Loebner Prize is awarded to the chatbot that does this task the best. It can be reiterated that chatbots are far reaching application and have the potential to be integrated in various domains. This work is motivated by the need to understand, analyze and catalog the existing work on chatbots in the academia as well as the industry alike. It can be postulated that chatbots can become virtual personal assistants that enable enhanced perception which is easily available to humankind on a daily basis. However, present-day chatbots are far from passing the Turing test, for which this competition was introduced.

Related Work

Chatbots have numerous real-world applications. Specifically, e-learning, marketing, medical diagnosis, cultural heritage, e-customer care services, task organizer are domains which make extensive use of chatbots. Moreover, their application is further pronounced when the activity takes place over the internet. E-learning chatbots can significantly contribute to providing interactive learning experience as well as individual attention to the improvement of each student. One solution for e-learning chatbot [9] starts with the basic NLP algorithm, Latent Dirichlet Allocation (LDA) which helps to process user's query, to remove stop words and extract keywords. Ontology is then built between learning concepts, depending on the course, lesson, topic, user etc. A chatbot for recommending tires has also been implemented using Petri Net [10]. It builds an ontology based on the knowledge domain of tires. Petri Net is formed by making use of user's responses to the type of vehicle (car, scooter, etc.), model number, year of manufacture etc. Each response is given weight and weighted sum of each context is calculated. If the weighted sum of a given context is less than the information the system already knows, context switching takes place so that repeated questions are not asked in a loop. Another usefulness of chatbots has been explored in the field of medical diagnosis [11]. The fact that previous clinical data of the patient is required for any such interaction is emphasized upon. Training of such a system has to be done on a regular basis for updated information regarding any disease and its diagnosis. Medical field is quite critical and has a long way to go in terms of chatbot based technology. Based on the tourist profile and the concept of context, chatbot has been developed which suggests various tourists' places, information about the place and services related to it. It can also suggest hotels nearby as well as famous dine in places. This chatbot is supposed to work as a tourist guide. The architecture has an inference engine at its core. This engine first analyses the text provided by the user and then generate useful reply using Context Dimension Tree (CBT) [12]. Knowledge acquisition is an important aspect of building a chatbot. Data filtered with respect to context, person, place etc. has to be acquired for chatbot to perform efficiently. Curious Cat [13] was designed to collect data from users by finding the right crowd, quality of conversations, consistency of replies, also known as crowdsourcing. This chatbot is further used as a personal conversation assistant. The academia as well as the industry has significantly progressed the usage and development of chatbots. These virtual assistants are set to become an indispensable tool in the foreseeable future.

Classical/Rule-Based Approaches

Classical approaches can be called as rule-based approaches as they set predefined rules to generate responses. These rules have grown more complex and sophisticated over time. These works very well when the domain of the conversation is closed i.e. the conversation is centered on a particular topic/task. But as the input becomes more natural or the domain moves to the open one, efficiency of rule-based approaches deteriorates. Writing rules can be done by a language designed for classical chatbots, AIML (Artificial Intelligence Markup Language), which is based on XML (eXtensible Markup Language).

Pattern Matching

Pattern Matching is one of the fundamental techniques of designing chatbots and is used in almost every chatbot to some extent. This method makes use of a prewritten set of rules and predefined templates to produce the response. ELIZA [14] was the first chatbot designed using this technique. Initially, it identifies the keywords in the text starting from left to right. Each keyword has a RANK/precedence associated with it. Then, the input string is decomposed in a predefined template. E.g. For input: I am sad, it takes 'sad' as a keyword and forms the reply- how long have you been 'sad'? ELIZA was a psychotherapist program and many people grew attached to it after it's invented. The challenges associated with this approach included the identification of the most important keyword and appropriate transformation rule. In addition, it does not take into account the previous history or context of the conversation which makes it look less natural.

Parsing

Parsing is the process of breaking down the input string to reveal its syntactic structure. The string is first divided into noun and verb phrase. Then, adjectives, articles, and nouns are recognized and a syntax tree is formed. Parsing helps validate a sentence's grammatical structure with respect to a language. Earlier, simple parsers were being used which could identify the keywords. For instance, 'take the food' and 'can you get the food' would both be parsed to 'take food'. This enables a chatbot with limited templates and patterns to generate the response for polymorphic input strings. Later chatbots use complete parsing techniques involved in processing natural language. This type consists of three levels of parsing which are syntax, semantics and pragmatic parsing [15]. Jabberwacky uses this technique for business circumstances where more control over conversational flow is required [16].

Markov Chain Models

Markov chain model, if described mathematically, is the model that describes the probability of present events on the basis of the state of previous events. It takes into consideration the probability with which a letter or word occurs within a dataset. It makes use of this probability distribution to choose the most likely words for a reply. The order of the Markov chain determines how many successive letters/words are to be taken as input. For a 0 order Markov chain, given a string khdddkhhdd, the letter k occurs with a probability of 2/10, h with 3/10 and d with 5/10. For order 1 Markov chain, it will also consider the previous element to compute the fixed probabilities.

Given a string "the black dog jumped into the pool". For an order 2 Markov chain 'the black' will result in 'dog', 'black-dog' will result in 'jumped' and so on for remaining words. If two results share different input then 0.5 probabilities will be assigned to both the input string.

The chatbot built on this method (HeX) used to generate a nonsense sentence that used to sound right, as a failback method [17] Another chatbot MegaHAL by the same scientist, used the entropy to determine the most likely word for a response out of many probable candidates [18].

$$I\left(\frac{w}{s}\right) = -log_2 P(\frac{w}{s}) \quad (1)$$

where w is the word following symbol sequence s

Semantic Nets (Ontologies)

Ontologies are a hierarchical structure of real-world concepts. Concepts are also called classes, which are the focus of most ontologies. Instances of various classes, when combined together along with the ontology, form the knowledge base. For example, a class of bread represents all bread. Further, they are divided into subclasses such as white bread and brown bread [19]. These classes can be connected to each other making a graph of hierarchy, where white and brown bread are subclasses of bread superclass. The classes can be connected on the basis of their logical relationship with each other. The properties of the class are defined by 'slots'. It may include bread's texture, color, company etc. Various 'facets' of the slots can also be defined. These describe the value type, cardinality, range of the slot etc. Its advantage lies in the fact that searching through the nodes can be done as well special reasoning rules can imply new responses. OpenCyc [20] and Wordnet [21] ontologies have been used in chatbots.

AIML

Artificial Intelligence Markup Language [22] is one of the technological advances dedicated to the development of chatbots. It is used for dialog modeling between a chatbot and human where the stimulus–response methodology is followed. Pattern Recognition and Matching Techniques form the basis of AIML. It is easy to implement as it is closely related to XML (eXtensible Markup Language) and the tags assist in making the task of dialog making it much simpler. Graphmaster which implements the pattern matching algorithm is responsible for managing the tree which is formed by storing the patterns of AIML. It provides efficient utilization of space as well as time. It is also highly reusable because of its simplicity as well as the availability of source code along with documentation.

Structure of an AIML tag is:

< command > ParametersList < /command >

where < command > is start tag and < /command > is closing tag. The most used tags are category, pattern, and template. The knowledge-base unit or commonly called dialogue is defined by category. The pattern defines the user's probable input and chatbot's response is defined in the template.

< category >
< pattern > how are you? < /pattern >
< template >
I am absolutely fine!
< /template >
< /category >

AIML also defines wildcards which are '_' and '*'. They replace a string or a part of the string. AIML gives high priority to categories which have wildcard within them and they are analyzed first.

< category >
< pattern > I love * < /pattern >
< template > I too love < star/ > . < /template >
< /category >

< srai > tag is also a powerful tag in AIML, as it has the ability to submit its own response as input to itself. Such a thing is useful when the user recursively talks about a particular topic, and this technique gives chatbot a chance to respond in the most natural way.

Wallace [23] created this XML dialect.

A.L.I.C.E [24] was the first chatbot based on AIML. The learning model used in ALICE is supervised one, i.e. it is being supervised by a person, the botmaster. After the initial design of ALICE, many other chatbots were built using AIML with further improvements.

Chatscript

Chatscript is the chatbot scripting language. It was developed by Bruce Wilcox in 2010. His chatbot 'Suzette' [25] won the 2010 Loebner Prize. Chatscript is basically an improvised version of AIML. Instead of searching for a matching category amongst thousands, chatscript searches for a related context. Such a context is called 'concept' and rules are defined within each concept. Concepts are nothing but a set of synonyms or words that are similar in some way. A concept of all pronoun, the noun can be created. Matching of each user input is done against the concepts preloaded into chatscript. Word-net Ontologies can be combined with chatscript to give better responses. The wildcards are also present in chatscript as in AIML. Apart from that, it also introduces the concept of variables, which can be used to store user-specific local information, which makes the conversation more natural and effective. Facts such as subject-verb-object triples can be created by chatscript and further stored in the tabular format. This table comes in handy while answering user input by simply querying into it.

Concept: ~ food (bread, juice, vegetable, fruits, pizza, burger, cold-drink)

S: (I love ~ pizza) Are you a foodie?

Structured Query Language (SQL) and Relational Database

Relational database (RDB) management system is used in the development of the chatbot. The primary objective behind using the database is to remember previous conversations and generate different replies even to the same questions posed at different interval of time. The most used RDB language is SQL. ViDi (Virtual Diabetes physician) [26] has been developed using this technique. This chatbot was specifically associated with the knowledge of diabetes disease. In this approach, forward and back pointers are maintained within the database also called extension and prerequisite variables. Whenever a response is generated, it is linked to another response/s based on the underlying knowledge base. These links are then used to generate new responses for each user input.

Language Tricks

It is often more natural to introduce concepts in a chatbot that are human-like. These may include deliberately committing a mistake in spelling or impersonating itself. Language tricks are often an additional technique used in the development of a chatbot. Some of the common language tricks are:

1) Typing errors and fake keystrokes: When a user types in an input, he/she usually examines the chatbot as it is typing the reply. It looks very human-like to fake backspaces and commits some spelling mistakes, which are some natural tendencies of humans.
2) Canned Responses: There are some patterns which the chatbot is unable to cover in its pattern matching algorithm. Such responses are hard-coded by the developer.
3) Personal History: Developers provide an identity to the chatbot to make it more convincing. The details about its birth, age, parents, preferences, stories are inculcated into it [17].

Neural-Network Based

Neural-network based chatbots have done away with the monotonous task of writing rules for each utterance-response pair. There are two ways in which neural network can output reply, either by producing from scratch (generative) or by retrieving from the large dataset (retrieval-based). Some hybrid approaches combining these two have also been introduced. The basic underlying structure/model employed in

all approaches has been discussed. Then a table (Table 1) is presented which covers the most recent work done on top of basic structures/models.

Recurrent Neural Network

The ability to consider previous conversations and context while generating a response is desirable for any conversational agent. The responses become monotonous if it only takes current input into account while forming a reply. Recurrent Neural Network (RNN) allows the chatbot to take as input the previous output, and come up with a more sensible reply. In other words, RNN allows the data to persist, unlike normal Neural Network.

In the above Fig. 2, A is a small part of the neural network and x_t is the input to it, it outputs ht. Since there is a loop forming in the network, it signifies the flow of data from output to input again. This loop makes the idea of RNN look unclear, but when we unroll the loop we will find a simple neural network that passes information from one network to the other.

RNNs have been used extensively for various purposes such as language translation, modeling speech recognition, image captioning etc. However, the unmodified version of RNN is not used much because it suffers from vanishing gradient problem [27].

Long Short Term Memory(LSTM) [28]

It becomes difficult for a simple RNN to remember information seen multiple steps ago when the unrolling steps increase too much. This is because the value of the gradient depends majorly on two factors which are weights and the activation function (basically their derivative). When either of them approaches to 0, the gradient vanishes with time. Activation Functions such as tanh and sigmoid makes the condition even worse as their derivative values are mostly close to 0.This is where LSTM comes into the picture and solves the problem of vanishing gradient. LSTM uses identity as its activation function whose derivative is 1 which prevents the backpropagated gradient to vanish. LSTM does this task with the help of 'gates'. Gates are the component which decides the information that will be allowed to pass through. The gates output the value between 0 and 1, deciding how much of each component should be let through. A value of 0 means not to let anything pass through and 1 means let everything pass. LSTM takes help from the input and forgets gates to control the flow of information from one network unit to the successor unit. These gates determine the network's state update mechanism. The output gate determines the output from the hidden layer.

The three gates together form a memory cell of LSTM. LSTM does the task of remembering, for instance, the gender of the subject, so that the chatbot can use 'his/her' depending on the previously remembered input. LSTM overcomes the problem of long-term dependencies. However, not all LSTM share the exact same structure. There are many variations of LSTM being proposed [29]. Another popular variant is Gated Recurrent Unit (GRU) [5]. In this architecture the input and forget gate are combined to form a single "update gate".

Seq2seq

One of the most effective techniques for machine translation [7], seq2seq can also be effectively applied to conversational modeling. The basic structure of a seq2seq model consists of two RNN as shown in Fig. 3. RNN is generally used in the form of LSTM or GRU. The objective is to calculate the conditional probability of p($y1$,$y2$,… yn'/$\times 1$,$\times 2$,…xn) where x and y represent the input and output sequences, respectively. Length of n and n' can vary. Seq2seq can easily allow such condition as two RNN are used for input and output sequences. The encoder-decoder mechanism is used. Firstly, the input sequence is subject to the first encoder RNN and a vector is produced as its output. Second RNN sets its initial state according to this vector. The output from this decoder RNN is then subjected to a suitable probability function. These two networks are trained together and back-propagation takes place and weights are adjusted accordingly.

Since seq2seq was developed primarily for machine translation, it takes source language sentence and converts it to vectors which represent word embeddings. The target language is decoded by the second RNN. In the case of chatbots, this technique can easily be used with slight modification by considering input sentence as the source language string and its response as the element for target language string.

Deep Seq2seq

Further improvement over seq2seq models can be done by joining multiple LSTM rather than just two. Better performance is expected out of such a model with deeper layers [30]. The most simple procedure to design such a model is to keep forwarding output from the previous layer to the next layer. The first encoder layer is fed with the input string. The encoder LSTM performs the task of conversion of each word to the vector. This output is fed to the next layer of encoder LSTM. Finally, the output from last encoder LSTM is passed to the first layer of decoder LSTM. Here again, the output is forwarded from one layer to the next. Finally, a suitable probability function is applied to get the target string.

Table 1 Design Techniques for chatbots along with evaluation metrics, corpus used and possible enhancement areas

Technique employed	References	Evaluation metrics	Corpus	Possible areas of enhancements
This paper proposes an encoder-decoder framework for conversational modeling. The attention mechanism is applied to the model, and beam search is used for decoding	Neural responding machine for short-text conversation [37]	Human annotation	Weibo is a popular Twitter-like microblogging service in China	Only for short text conversation
This paper focuses on generating context-sensitive responses by encoding past information using embedding based model. This work utilizes Recurrent Neural Network Language Model (RLM) architecture and tons of features are added on top of it	A neural network approach to context-sensitive generation of conversational responses [33]	BLEU, METEOR, Human Evaluation	Response-triplets using Twitter FireHose	Bag of words model is used which does not take into account the order within context and message
HRED architecture is used in which RNN encodes the input utterances. These encoded vectors are used by context RNN as context vector. Finally, GRU is used to encode the structure of input utterance seen so far. The decoder, takes as input, the output of context RNN and with the help of beam search, produces output	Building end-to-end dialogue systems using generative hierarchical neural network models [38]	Perplexity, Word Error-Rate	MovieTriples dataset	Generic Responses of 'I don't know' are frequent
Maximum Mutual Information(MMI) has been used in place of likelihood of output, as objective function	A diversity-promoting objective function for neural conversation models [39]	Multireference BLEU, human evaluation	Twitter Conversation Triple Dataset,OpenSubtitles dataset	Factors such as grounding, persona and intent have not been covered
Apart from encoder, decoder LSTM structure, Intention structure is also included	Attention with intention for a neural network conversation model [40]	Perplexity	In-house helpdesk chat service	Intention-specific
Persona/artificial agent has been introduced by capturing the speaking style and background	A persona-based neural conversation model [41]	Perplexity, BLEU, Human Judgment	Twitter dataset, dataset from TV series	Not able to capture mood, emotions at a particular point of time
Overcomes the problem of chatbots being passive, i.e. computer takes the initiative and introduces new content. When a stalemate is detected using keywords, named entity recognition is applied on previous conversations. Retrieval and ranking based system	StalemateBreaker : a proactive content-introducing approach to automatic human–computer conversation [6]	Mean Average Precision (MAP), nDCG(normal Discounted Cumulative Gain), p@1	(Chinese) forums, micro blog websites, and community question-answering platforms	Retrieval based system

Table 1 (continued)

Technique employed	References	Evaluation metrics	Corpus	Possible areas of enhancements
VHRED model is introduced which extends the HRED model by augmenting latent variable at the decoder. The training step is done by maximizing variational lower bound on log likelihood	A hierarchical latent variable encoder-decoder model for generating dialogues [42]	Embedding-based metrics(Greedy, Average, Extreme),	Twitter Dialogue Corpus, Ubuntu Dialogue Corpus	Longer utterances generated every time, even when short replies are expected as well as suitable
Reinforcement learning has been applied in conjugation with seq2seq model	Deep reinforcement learning for dialogue generation [43]	Human judgment	OpenSubtitles dataset	Rewards are heuristic and hence does not lead to an ideal conversation
Incorporated attention in HRED model. IDF term is used in objective function	An attentional neural conversation model with improved specificity [34]	BLEU, perplexity	helpdesk chat service dialogues	Larger training data required for better results
This paper suggests that more the number of previous conversation turns, the better the response generated	Neural discourse modelling of conversations [44]	perplexity, discourse analysis metrics	OpenSubtitles dataset	Increasing the value of N has a trade-off on time and resources
Input to decoder goes through two neural networks, one is encoder network,other is CNN(for learning topic distribution)	Neural contextual conversation learning with labeled question-answering Pairs [45]	perplexity	Two popular question–answer websites:Baidu Tieba and Douban	Perplexity is low for shorter sentences
First, a keyword is chosen using pointwise mutual information,then the reply is generated by going forward and backward using two RNN	Sequence to backward and forward sequences: a content-introducing approach to generative short-text conversation [46]	Human Evaluation, length, entropy	Baidu Tieba forum	Shorter replies than seq2seq
A response is retrieved and fed to generative part of the model. Then the resultant reply is compared with that of retrieved one by reranking them	Two are better than one : an ensemble of retrieval- and generation-based dialog systems [8]	Human Evaluation, BLEU, length, entropy	Massive online forums, microblogs, and question-answering communities,	No mention of results on goal-oriented system
Generator-Discriminator model is introduced first time for NLP in this paper. Generator is seq2seq model and discriminator is used to distinguish dialogs whether they are human or machine generated	Adversarial learning for neural dialogue generation [47]	evaluator reliability error(ERE), Human evaluation	OpenSubtitles dataset	This model does not perform very well if there is less discrepancy between generated and reference sequences
Responses are based on contextual history as well as facts from knowledge base (Amazon, Wikipedia) on top of seq2seq model. Beam search is used along and reranking is done on MMI	A knowledge-grounded neural conversation model [48]	perplexity, BLEU, human evaluation	Twitter, Foursquare	Since it deals with facts as a knowledge base, some facts can be irrelevant or contradictory

Table 1 (continued)

Technique employed	References	Evaluation metrics	Corpus	Possible areas of enhancements
RNN is used. Utterance is processed in three ways: using bag of words, embedding and entity extraction, are passed to RNN	Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning [49]	Dialog success rate	bAbI dialog dataset	Model need to be deployed in live dialog system
Conversation is represented using dialog context, response utterance and latent variable. Knowledge-guided Conditional Variational AutoEncoder (CVAE) deployed	Learning discourse-level diversity for neural dialog models using conditional variational autoencoders [50]	BLEU, cosine distance, Dialog Act Method	Switchboard (SW) 1 Release 2 Corpus	Various improvements suggested by author like using deep neural network learning powers, and considering linguistic phenomena
Common-sense knowledge base is integrated with retrieval-based models	Augmenting end-to-end dialog systems with commonsense knowledge [51]	recall @ k	Twitter dataset	only for retrieval based scenario
Utterance-level LSTM is used. Two strategies employed on top of it. One is policy network and another is Reinforcement Learning	End-to-end optimization of task-oriented dialogue model with deep reinforcement learning [52]	Human evaluation	Restaurant and movie booking domain dataset	Only updating policy network results in lesser performance improvement as compared to Reinforcement learning
Seq2seq model is augmented with memory network that help encode personas(information about themselves)	Personalizing dialogue agents: i have a dog, do you have pets too? [53]	perplexity, F1 score, next utterance classification loss	Crowd-sourced dataset	Trained on persona-chat, can be done in a way where model learns and gains persona from chat history itself and remember that
The degeneration problem of VAEs has been solved using utterance drop regularization	A hierarchical latent structure for variational conversation modeling [54]	Negative log-likelihood embedding-based metrics,human evaluation via Amazon Mechanical Turk (AMT)	Cornell Movie Dialog Corpus, Ubuntu Dialog Corpus	Overfitting in case of Cornell Movie Dialog Dataset
This is a VAE based approach with discrete latent variables. Two models suggested, one DI-VAE (Recognition and Generator network), other is DI-VST (Discrete variational skip-thought)	Unsupervised discrete sentence representation learning for interpretable neural dialog generation [55]	Perplexity, KL divergence, Mutual information (between input data and latent variables)	Penn Treebank, Stanford Multi-Domain Dialog, Daily Dialog and Switchboard	Better context based latent actions learning is possible
GAN is being trained within the latent variable space. DialogWAE with Gaussian mixture network performs better than previous models for dialog generation	DialogWAE : multimodal response generation with conditional wasserstein auto-encoder [56]	BLEU, BOW Embedding, distinct,human evaluation	Dailydialog and Switchboard	Intra-distinct scores not better because of long responses
Combination of HRED and GAN, along with teacher forcing	Multi-turn dialogue response generation in an adversarial learning framework [57]	Perplexity, BLEU, ROUGE and Distinct n-gram scores	Movie Triples corpus, Ubuntu Dialogue corpus	No human evaluation
Diversity in responses has been increased using adversarial training. CNN encoder and LSTM decoder is used	Generating informative and diverse conversational responses via adversarial information maximization [58]	BLEU, ROGUE, Embedding-based metrics(Greedy, Average, Extreme), Diversity metrics(Dist-1, Dist-2,Entropy)	Reddit, Twitter	Distributional discrepancy between ground-truth responses and responses generated has not yet been covered

Table 1 (continued)

Technique employed	References	Evaluation metrics	Corpus	Possible areas of enhancements
The encoder part of the transformer has been used in the model. Masking of words procedure has been used, where it can mask existing words or replacing them with random words	BERT: pre-training of deep bidirectional transformers for language understanding [59]	General Language Understanding Evaluation (GLUE)	BooksCorpus, Wikipedia	Linguistic phenomena still to be captured

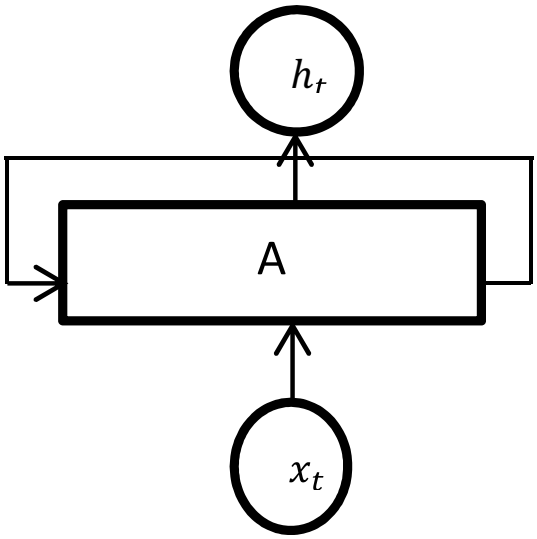

Fig.2 Recurrent Neural Network

Evaluation Methods

Mostly used metrics for evaluation of a chatbot are BiLingual Evaluation Understudy (BLEU) [31] and perplexity [32], METEOR (Metric for Evaluation of Translation with Explicit ORdering) [33] which were originally meant for machine translation methods. These measures are used for conversational modeling at various places [30, 34, 35].

BLEU measures the similarity between generated text and the expected response. A score of 1.0 represents a perfect match whereas 0.0 represents a perfect mismatch. It measures the adequacy and fluency of a generated text by counting the words which match with the expected response. Matching of words takes place for every word, in pair, in triplets and so on, also called n-grams. For $n=1$, it would consider a single token (unigrams), for $n=2$ a word pair is considered (bigrams) and so on. Order of grams (words) is not significant in this method.

E.g. He is the only son of Great Odin. (Expected response).

Great Odin has only one child. (Generated response).

'Only' is the unigram and 'Great Odin' is the bigram that matches in both the sentences.

To overcome some of the limitations of BLEU metric, authors have used METEOR which is very much similar to BLEU, with the added functionality of synonym matching and mapping between generated and expected response. It matches the exact words in the two sentences; each word in expected response is mapped to another word in the generated response. Synonyms are found for mismatched words. After matching the unigrams, the score is computed based on unigram precision and recall.

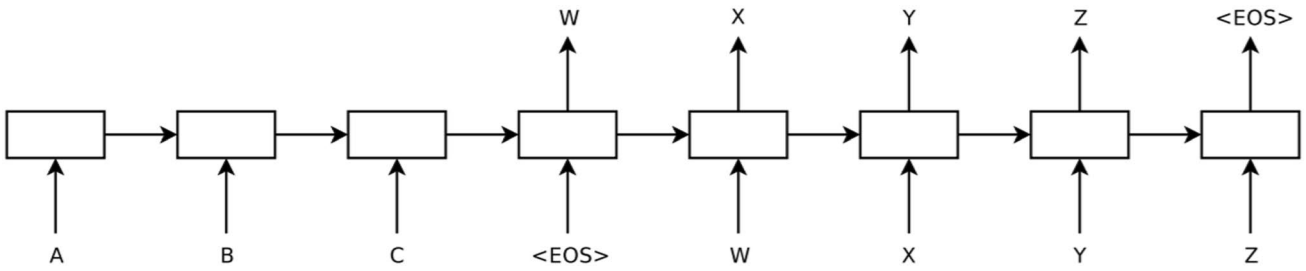

Fig. 3 Sequence to sequence model [7]

Perplexity defines the goodness of a probability model to predict a test data. Perplexity is exponentiation of entropy. After the training of the model, the test set can be used to compute the perplexity. If a model 'q' exists, perplexity is given by:

$$2^{\frac{-1}{N}\sum_{i=1}^{N} log_2 q(x_i)} \quad (2)$$

X_i is the test set or input words.

N is the length of the sentence.

Model performs better when perplexity is less.

The evaluation methods for a conversational bot still remain a question in the open domain. This is because the effectiveness of a chatbot can only be evaluated in a real-time domain. The task of evaluating a chatbot is subjective which deals closely with human judgement. Metrics like BLEU, METEOR, and perplexity have been extensively used but the general consensus remains that one cannot completely encompass user experience using traditional mathematical indicators. User experience has been measured with the following metrics: user engagement, coherence, domain-coverage, depth of conversation etc [36].

User engagement is measured by the duration of chat between human and chatbot. More number of turns in conversation might mean that the chatbot is able to provide answers so as to keep the user engaged. Coherence is measured by the relevancy of the reply generated. This is generally a hard objective to reach in an open ended conversation but is extremely important as well. E.g. If a user is talking about Politics and gets a response unrelated to it, like sports, it would be considered a weakly coherent response. A task-oriented chatbot is domain-specific, whereas an open domain conversational agent is expected to deal with multiple domains. In the case of multi-turn conversation, it is important the chatbot is able to converse about a topic in some depth, as it happens with humans.

So, the best method to evaluate a chatbot is to get it rated by a human being. He/she can decide whether the responses generated were meaningful and natural. The grammar, effectiveness, and naturalness of a chatbot can only be judged truly by a human. For a task-oriented chatbot, the user can be asked whether they feel satisfied with the responses or whether the chatbot was able to answer their queries.

The work was conducted systematically by bifurcation of the paper search space into different relevant domains. The first domain was taken as Rule-based and other is Neural Network based. Among each domain, chronological ordering was followed to build a stronger understanding of the works with respect to the evolution of chatbots.

The table presented outlines the recent developments in the field. Many variations of encoder-decoder networks have been used. Deep learning models have been used extensively like HRED, GAN, VAE etc.

Discussion and Future Work

The backbone of conversation modeling is encoder-decoder model. This model was designed for Neural Machine Translation (NMT). However, conversation modeling is altogether a complex task to be done using this model. This is because encoder-decoder model assumes one single reply for a given input. This is not true for conversation agents as a natural response can vary for the same input at different time and condition. The encoder-decoder model averages out the utterance-response pair. This is why it was noted in many papers that generic responses such as 'I don't know' have been produced by different models. Also, evaluating these models has long been a challenge posed to AI practitioners. Since quantitative evaluation metrics such as BLEU and perplexity are far from human judge evaluation, especially for chatbots. Other metrics have also been introduced in several papers but no standard method exists for chatbots till now.

As for future work, there are various areas which still need to be explored in the field of conversation modeling.

- The objective function: Log likelihood and MMI have been majorly been used as objective functions. Log likelihood measures the most probable response for a given utterance. To take previous conversations and context probabilities, experiments can be done on optimizing the objective function.

- Persona development: Many authors agree to the fact that conversations look more natural when they have imbibed personalities in speaker and addressee. This task, however, ought to be done by the model by understanding the speaking style and mood of the person. Work has been done in this area, but still need to consider various other parameters as well.
- Two-sided conversation: It is true that most chatbots are made to reply to the given utterance, but this makes the conversation one-sided. Hence, it is important for the chatbot to come up with topics that interests the person it is talking to. This again can be done by encoding huge amount of conversation history and persona building.

Conclusion

Chatbots have become an integral part of our day to day life. A great deal of effort is employed to make it talk like a human. Nowadays, chatbot is a part of almost every application which deals with activities like ordering clothes, food, electronic appliances and so on. They are also used to book tickets, appointments, shows, or any transactional activity. Businesses use chatbot to solve customer's problem by suggesting frequently asked questions and try to make the conversation interactive. If the customer is not satisfied, human intervention takes place in most cases. This review of chatbots presented gives a clear picture of the approaches that can be deployed in the development of a chatbot. Mostly the vanilla versions are presented which can be further manipulated and improved. Starting from fundamental approaches like pattern matching, parsing, semantics Nets to deep neural network-based approaches such as RNN, LSTM, have been cited with their respective chatbot example While going through the review, reader gets the idea of how chatbots evolved with time. Modern day chatbots still use those played out but powerful techniques. More and more chatbots these days are making use of neural network-based approaches, but keeping the advantageous elements of non-AI based methods. This observation is visible in this review that includes the latest work done in the field of conversational agents. However, it is quite clear that conversational bots, as of now, are far from passing the Turing test. Still, on the road to improvement, various quantitative and qualitative metrics to determine the efficiency of a chatbot have been discussed. The paper is concluded with the most recent work done in the field of conversational modeling. It is hoped that this work shall propel the research community with a better understanding of chatbots.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Turing BAM. Computing machinery and intelligence. In: Parsing the turing test. Springer, Dordrecht; 2009. p. 23–65.
2. Serban IV et al. A deep reinforcement learning chatbot. arXiv preprint arXiv: 1709. 02349v [cs . CL] 2017; 1–40.
3. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems; 2012. p. 1097–105.
4. Havrylov S, Titov I. Emergence of language with multi-agent games: learning to communicate with sequences of symbols. In: Advances in neural information processing systems; 2017, p. 2149–59.
5. Van Merri B, Fellow CS. Learning phrase representations using RNN encoder—decoder for statistical machine translation. arXiv preprint arXiv: 1724–1734. 2014.
6. Li X, Mou L, Yan R, Zhang M. StalemateBreaker : a proactive content-introducing approach to automatic human-computer conversation. arXiv preprint arXiv:1604.04358. 1:2845–2851.
7. Sutskever I, Vinyals O, Le QV. Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. p. 3104–12.
8. Song Y, Yan R, Li X, Zhao D, Zhang M. Two are better than one : an ensemble of retrieval- and generation-based dialog systems. arXiv preprint arXiv:1610.07149. 2016 ; 1:1–11.
9. Clarizia F, Colace F, Lombardi M. Chatbot : an education support system for student, vol. 1. Berlin: Springer; 2018.
10. Colace F, De Santo M, Pascale F, Lemma S, Lombardi M. BotWheels: a petri net based chatbot for recommending tires. In: Data; 2017. p. 350–8.
11. Edwards BI, Muniru IO, Cheok AD. Robots to the rescue: a review of studies on differential medical diagnosis employing ontology-based chat bot technology. Preprints. 2016. https://doi.org/10.20944/preprints201612.0027.v1.
12. Casillo M, Clarizia F, Aniello GD, De Santo M, Lombardi M, Santaniello D. CHAT-Bot: a Cultural Heritage Aware TellerBot for supporting touristic experiences. Pattern Recognit Lett. 2020;131:234–43.
13. Witbrock M. Conversational crowd based and context aware knowledge acquisition chat bot. 2016; 239–252.
14. Weizenbaum J. ELIZA—a computer program for the study of natural language communication between man and machine. Commun ACM. 1966;9(1):36–45.
15. Ahmad S. Tutorial on natural language processing. Artif Intell. 2007;810:161.
16. "https://www.jabberwacky.com/j2about." [Online]. Available: https://www.jabberwacky.com/j2about. Accessed21 July 2019.
17. Bradeško L, Mladenić D. A survey of chabot systems through a loebner prize competition. Res Net. 2012;2:1–4.
18. Hutchens JL, Alder MD. Introducing MegaHAL II I ! II. Computer (Long. Beach. Calif). 2000;1998:271–4.
19. Noy NF, McGuinness DL. Ontology development 101: a guide to creating your first ontology. Stanford: Stanford Knowl Syst Lab; 2001. p. 25.
20. Lenat DBCYC. A large-scale investment in knowledge infrastructure. Commun ACM. 1995;38(11):33–8.
21. Zubaide HAl, Issa AA. OntBot: ontology based ChatBot. 2011 4th Int Symp Innov Inf Commun Technol. ISIICT'2011. p 7–12,

22. BrunoG, De Aguiar RV, Barbosa GDO, Botelho WT, Pimentel E. A RTIFICIAL I NTELLIGENCE M ARKUP L ANGUAGE : A B RIEF T UTORIAL.
23. Wallace R. The elements of AIML style. Alice AI Foundation 139; 2003.
24. Wallace RS. The anatomy of ALICE. In: Parsing the turing test. Dordrecht: Springer; 2009. p. 181–210.
25. Wilcox B, Wilcox S. Suzette, the most human computer. Agent's Processing, Cognition. 2010. https://www.chatbots.org/images/uploads/research_papers/9491.pdf.
26. Razak LT. Extension and prerequisite : an algorithm to enable relations between responses in chatbot technology abbas saliimi lokman and jasni mohamad zain faculty of computer systems and software engineering. J Comput Sci. 2010;6(10):1212–8.
27. Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int J Uncertainty Fuzziness Knowl Based Syst. 1998;6(2):107–16.
28. Cascade-correlation R, Chunking NS. Long short term memory. Neural Comput. 1997;9(8):1–32.
29. Greff K, Srivastava RK, Koutnik J, Steunebrink BR, Schmidhuber J. LSTM: a search space odyssey. IEEE Trans Neural Networks Learn Syst. 2017;28(10):2222–32.
30. Vinyals O, Le Q. A neural conversational model. 2015. arXiv :1506.05869.
31. Papineni K, Roukos S, Ward T, Zhu W. BLEU: a method for automatic evaluation of machine translation. 2002; 311–318.
32. Manning C, Schutze H. Foundations of statistical natural language processing. MIT press; 1999.
33. Galley M, Auli M, Brockett C, Ji Y, Mitchell M, Gao J. A neural network approach to context-sensitive generation of conversational responses. arXiv preprint arXiv:1506.06714. 2015.
34. Peng B, Zweig G. An attentional neural conversation model with improved specificity. arXiv preprint arXiv:1606.01292. 2016.
35. Zhao T, Lu A, Lee K, Eskenazi M. Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. arXiv preprint arXiv:1706.08476. 2017; 27–36.
36. Gabriel R et al. On evaluating and comparing conversational agents. Nips. 2017; 1–10.
37. Shang L, Lu Z, Li H. Neural responding machine for short-text conversation. 2015; 1577–1586.
38. Serban IV, Sordoni A, Bengio Y, Courville A, Pineau J. Building end-to-end dialogue systems using generative hierarchical neural network models. 2015,
39. Gao J. A diversity-promoting objective function for neural conversation models. 2015.
40. Zweig V. Attention with Intention for a Neural Network Conversation Model. 2015; 1–7.
41. Li J, Galley M, Brockett C, Spithourakis GP, Gao J, Dolan B. A persona-based neural conversation model. 2016.
42. Serban IV, Sordoni A, Lowe R, Charlin L, Pineau J. A hierarchical latent variable encoder-decoder model for generating dialogues. 2016.
43. Li J, Monroe W, Ritter A, Galley M, Gao J, Jurafsky D. Deep reinforcement learning for dialogue generation. 2016.
44. Pierre JM, Butler M, Portnoff J, Aguilar L. Neural discourse modelling of conversations. 2016; 5(6):1–8.
45. Xiong K, Cui A, Zhang Z, Li M. Neural contextual conversation learning with labeled question-answering pairs. 2014, 2016.
46. Mou L, Song Y, Yan R, Li G, Zhang L, Jin Z. Sequence to backward and forward sequences: a content-introducing approach to generative short-text conversation. 2016.
47. Li J, Monroe W, Shi T, Jean S, Ritter A, Jurafsky D. Adversarial learning for neural dialogue generation. 2017.
48. Ghazvininejad M et al. A knowledge-grounded neural conversation model. 2017.
49. Williams JD, Asadi K, Zweig G. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. 2017.
50. Zhao T, Zhao R, Eskenazi M. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. 2017.
51. Young T, Cambria E, Chaturvedi I, Huang M, Zhou H, Biswas S. Augmenting end-to-end dialog systems with commonsense knowledge. 2017.
52. Liu B, Tur G, Hakkani-Tur D, Shah P, Heck L. End-to-end optimization of task-oriented dialogue model with deep reinforcement learning. 2017; 1–6.
53. Zhang S, Dinan E, Urbanek J, Szlam A, Kiela D, Weston J. Personalizing dialogue agents: i have a dog, do you have pets too?. 2018.
54. Kim G. A hierarchical latent structure for variational conversation modeling. 2018.
55. Zhao T, Lee K, Eskenazi M. Unsupervised discrete sentence representation learning for interpretable neural dialog generation. 2018.
56. Gu X, Cho K, Ha J, Kim S. DialogWAE : multimodal response generation with conditional wasserstein auto-encoder. 2018; 1–10.
57. Olabiyi OO, Salimov A, Mueller ET. Multi-turn dialogue response generation in an adversarial learning framework. 2018.
58. Zhang Y, Gan Z, Brockett C. Generating informative and diverse conversational responses via adversarial information maximization. Nips. 2018.
59. Devlin J, Chang M-W, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. 2018.