Recognizing Textual Entailment in Non-english Text via Automatic Translation into English

Partha Pakray[1], Snehasis Neogi[1], Sivaji Bandyopadhyay[1], and Alexander Gelbukh[2]

[1] Computer Science and Engineering Department,
Jadavpur University, Kolkata, India
[2] Center for Computing Research, National Polytechnic Institute,
Mexico City, Mexico
{parthapakray,snehasis1981}@gmail.com,
sbandyopadhyay@cse.jdvu.ac.in
www.gelbukh.com

Abstract. We show that a task that typically involves rather deep semantic processing of text—being recognizing textual entailment our case study—can be successfully solved without any tools at all specific for the language of the texts on which the task is performed. Instead, we automatically translate the text into English using a standard machine translation system, and then perform all linguistic processing, including syntactic and semantic levels, using only English language linguistic tools. In this case study we use Italian annotated data. Textual entailment is a relation between two texts. To detect it, we use various measures, which allow us to make entailment decision in the two-way classification task (YES / NO). We set up various heuristics and measures for evaluating the entailment between two texts based on lexical relations. To make entailment judgments, the system applies named entity recognition module, chunking, part-of-speech tagging, *n*-grams, and text similarity modules to both texts, all those modules being for English and not for Italian. Rules have been developed to perform the two-way entailment classification. Our system makes entailment judgments basing on the entailment scores for the text pairs. The system was evaluated on Italian textual entailment data sets: we trained our system on Italian development datasets using the WEKA machine learning toolset and tested it on Italian test data sets. The accuracy of our system on the development corpus is 0.525 and on the test corpus is 0.66, which is a good result given that no Italian-specific linguistic information was used.

Keywords: Recognizing textual entailment, *n*-grams, text similarity, machine translation, cross-lingual textual entailment.

1 Introduction

Recognizing Textual Entailment (RTE) is one of recent challenges of Natural Language Processing (NLP) [1]. Textual entailment is defined as a directional relationship between pairs of text expressions, denoted by T—the entailing "Text", and H—the entailed "Hypothesis". T entails H if the meaning of H can be inferred from the meaning of T, as would typically be interpreted by people.

I. Batyrshin and M. González Mendoza (Eds.): MICAI 2012, Part II, LNAI 7630, pp. 26–35, 2013.

For example, the following text: T = *John's assassin is in jail* entails the following hypothesis: H = *John is dead*. Indeed, if there exists one's assassin, then this person is dead. On the other hand, the text T = *Mary lives in Europe* does not entail a hypothesis H = *Mary lives in London*.

RTE is useful for many NLP tasks. For example, in text summarization (sometimes denoted by SUM) a summary of a document should be entailed by its contents; paraphrases can be seen as mutual entailment between the two expressions; in Information Extraction (IE), the extracted information should also be entailed by the documents; in Question Answering (QA), the answer obtained for a question must be entailed by the supporting snippets of text.

There exist a number of Recognizing Textual Entailment evaluation initiatives. There have been held seven Recognizing Textual Entailment (RTE) competitions: RTE-1 in 2005 [2], RTE-2 [3] in 2006, RTE-3 [4] in 2007, RTE-4 [5] in 2008, RTE-5 [6] in 2009, RTE-6 [7] in 2010, RTE-7 [8] in 2011. In 2010, Parser Training and Evaluation using Textual Entailment event [9] was organized in frame of SemEval-2.

In 2011, Recognizing Inference in Text (RITE) was organized by NTCIR-9.[1] In 2012, Cross-lingual Textual Entailment for Content Synchronization (CLTE)[2] track was organized in frame of SemEval-2012. Gradual advances and previous versions of the present work have been presented at RTE-5, RTE-6, RTE-7, SemEval-2 Parser Training and Evaluation using Textual Entailment Task, RITE, and SemEval-2012: Cross-lingual Textual Entailment for Content Synchronization.

In this paper, we report the results obtained with an improved version of our system. This system uses a chain of NLP modules in order to obtain a wide variety of features of both text T and hypothesis H, varying from *n*-gram based to syntactic and semantic levels.

The contribution of this paper consists in suggesting that certain tasks—in this case the recognizing textual entailment task as a case study—that involve deep language processing can be accomplished without the use of any tools or techniques specific for the given language.

Namely, we use a pivot language approach: our text processing modules work with English language data; the input texts in any language are translated into English using a standalone machine translation system. Thus, we show that machine translation can be used to successfully perform the RTE task in any language or even when T and H are in different languages.

For evaluation, in this work we use the EVALITA Textual Entailment data sets. EVALITA 2009[3] was an evaluation campaign of both Natural Language Processing and speech technologies for Italian language. The EVALITA Textual Entailment task[4] consisted in detection of inferential relationships between pairs of short texts.

The work is organized as follows. Our two-way textual entailment recognition system architecture is presented in Section 2. Section 3 describes feature extraction, to be used with the WEKA toolset [10]. The experimental results on the development and test data sets are given in Section 4. Finally, conclusions are drawn in Section 5.

[1] http://artigas.lti.cs.cmu.edu/rite/Main_Page
[2] http://www.cs.york.ac.uk/semeval-2012/task8/
[3] http://www.evalita.it/2009
[4] http://www.evalita.it/2009/tasks/te

2 System Architecture: A Machine Learning Approach

The EVALITA datasets used to test our system are available in the Italian language. The task that has been proposed for EVALITA is basically a binary classification textual entailment problem: a system should predict whether the text T in the text pair entails or not the corresponding hypothesis H.

We explore in this paper a machine learning based approach for this EVALITA task. Our system generates various lexical matching scores calculated over the development dataset are used to train the model along with the target class. Specifically, the system includes such components as the preprocessing module, lexical similarity module, and text similarity module. The lexical similarity module is in turn divided into sub-modules such as POS matching, chunk matching, and named entity matching.

This trained model was then used to predict the classification of unseen text pairs in the test dataset. The WEKA machine learning toolset is used to classify and predict the classification of text pairs. As the pairs are available in Italian language, our system uses pivot language approach: it applies the textual entailment module after automatically translating the text pair into English.

Figure 1 shows our system architecture, where the text and hypothesis sentences are translated into English.

2.1 Pre-processing

The system extracts the T (text) and H (hypothesis) pair from the EVALITA task data. The text and hypothesis pair is available there in the Italian language. Microsoft Bing translator API[5] for Bing translator (the file `microsoft-translator-java-api-0.4-jar-with-dependencies.jar`) was used to translate the T and H text sentences into English.

The translated Text and Hypothesis sentences were then passed through stop-word removal and co-reference resolution modules.

Stop-Word Removal. This module removes stop-words listed in a pre-defined stop-word list from the T and H sentences.

Co-reference Resolution. Co–reference chains are evaluated in the datasets before passing the text to the RTE module. The objective is to increase the entailment score by substituting the anaphors with their antecedents.

A word (often a pronoun) or phrase in the sentence can be used to refer to an entity introduced earlier or later in the discourse. The description that introduces the entity and all expressions that refer to it are said to have the same referent; this phenomenon is called co-reference.

We distinguish between two types of co-reference. When the reader must look back to the previous context to find the referent, then the co-reference is called

[5] `http://code.google.com/p/microsoft-translator-java-api/`

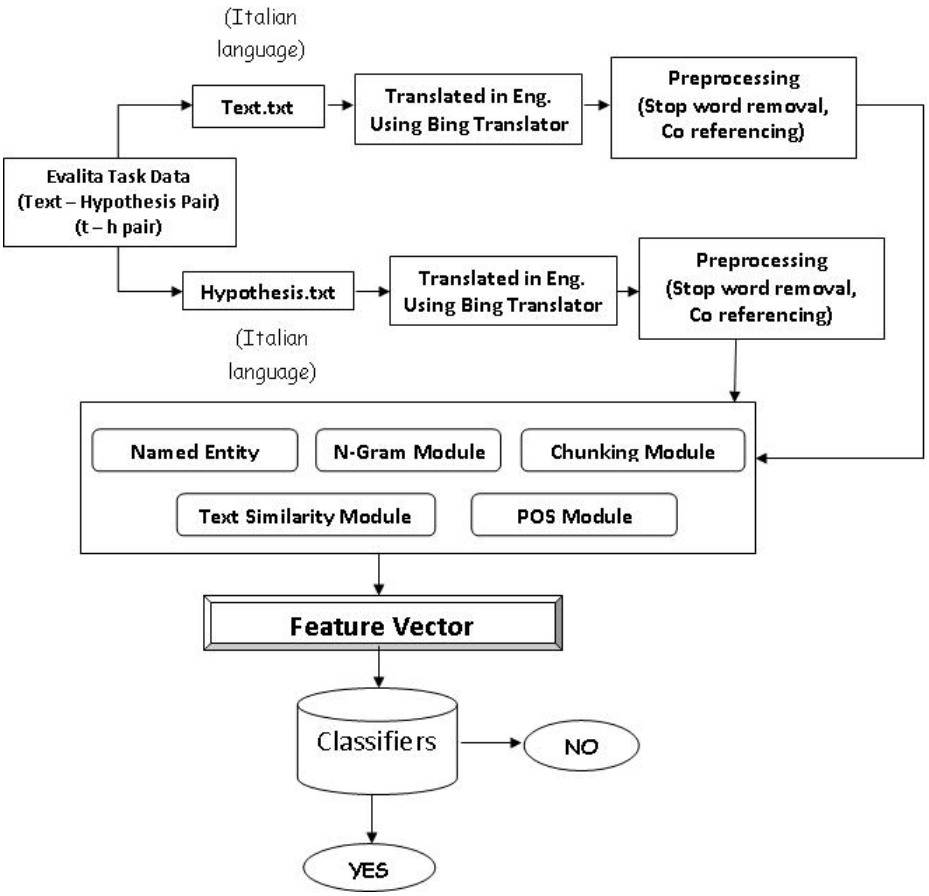

Fig. 1. Architecture of our system

anaphoric reference. When the reader must look forward to find the referent, then it is called cataphoric reference.

To address this problem we used a tool called JavaRAP[6], which is a Java-based implementation of the Resolution of Anaphora Procedure (RAP), an algorithm developed by Lappin and Leass [11]. We observed, however, that co-referential expressions are very rare in the sentence based paradigm.

2.2 Lexical Based Textual Entailment (TE) Recognition Module

Text–Hypothesis pairs are the inputs to the system. The overall TE module is a collection of several lexical-based sub-modules. Each sub-module produces a lexical matching score that is used to develop a training model.

[6] http://aye.comp.nus.edu.sg/~qiu/NLPTools/JavaRAP.html

N-gram Matching Module. The *n*-gram matching basically measures the percentage of unigrams, bigrams, and trigrams of the Hypothesis that are also present in the corresponding Text. Note that both the Text and the Hypothesis have been previously pre-processed with stop-word removal, so that the *n*-grams do not contain any stop-words.

The scores for different values of *n* are simply combined to get an overall *n*-gram matching score for a particular pair.

Chunk Similarity Module. This sub-module of our system evaluates the key NP chunks of both text and hypothesis, that are identified using the NP Chunker v1.1[7] tool. Then, our system checks the presence of NP chunks of the hypothesis in the corresponding text.

The system calculates the overall value for the chunk matching, i.e., the number of NP chunks of the text that match with NP chunks of the hypothesis. If the chunks are not similar in their surface form, then our system applies WordNet-based matching for the words: if they match in WordNet synsets information, then the chunks are considered similar.

WordNet is one of most important resource for lexical analysis. The WordNet 2.0 has been used for WordNet-based chunk matching. The API for WordNet Searching (JAWS),[8] an API that provides Java applications with the ability to retrieve data from the WordNet database, was used.

Text Distance Module. The system takes into consideration a wide variety of text similarity measures calculated over the each T–H pair. These text similarity measures are summed up together to produce the total final score for a particular text pair, which is used for the classification decision.

Specifically, the following well-known text similarity measures are used in our system:

- Cosine Similarity
- Levenshtein Distance
- Euclidean Distance
- Monge–Elkan Distance
- Needleman–Wunch Distance
- Smith–Waterman Distance
- Block Distance
- Jaro Similarity
- Matching Coefficient Similarity
- Dice Similarity
- Overlap Coefficient
- Q-grams Distance

[7] http://www.dcs.shef.ac.uk/~mark/phd/software/
[8] http://lyle.smu.edu/~tspell/jaws/index.html

Named Entity Matching. This process is based on the detection and matching of named entities (NEs) in the T–H pair. The Stanford Named Entity Recognizer (NER) was used to tag the named entities in both text and hypothesis. The system simply calculates the number of the Hypothesis's NEs that are present in the Text. A score is associated with the matching as follows:

$$\text{NE\_Match} = \frac{\text{Number of common NEs in Text and Hypothesis}}{\text{Number of NEs in Hypothesis}}.$$

Part-of-Speech Matching. This module basically deals with matching the common part of speech (POS) tags between the T and H. The Stanford POS tagger was used to tag words with the parts of speech in both the Text and the Hypothesis. The system calculates the number of the verb and noun POS words in the Hypothesis that match those in the Text. A score is associated with the number of patched POSs as follows:

$$\text{POS\_Match} = \frac{\text{Number of verb and noun POSs in Text and Hypothesis}}{\text{Total number of verb and noun POSs in Hypothesis}}.$$

3 Feature Extraction

The system-generated matching scores were fed in the training module of the WEKA machine learning tool to develop a classification model. This model is used to predict the presence or absence of entailment in the untagged text pair in the test set of the EVALITA task.

The main motivation to introduce a machine learning approach in this EVALITA task is that the Textual Entailment task can be considered as a classification problem. Different measures applied to the Text–Hypothesis pair can be used as a feature vector for the classifier. In this architecture we used lexical similarities as the feature vector. Naïve Bayes classifier was used to predict the outcome for unseen pairs. Figure 2 illustrates the concept of machine learning for the classification of textual entailment problem.

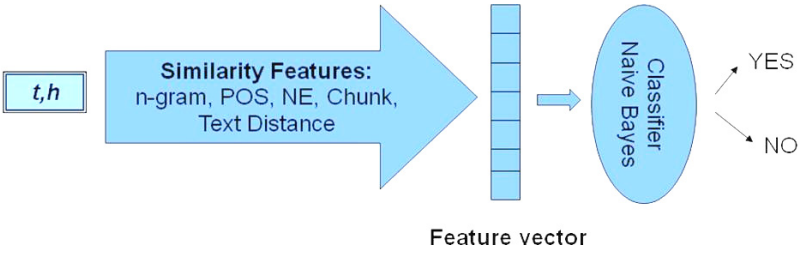

Fig. 2. Machine Learning Classification

The Naïve Bayes algorithm is a classification algorithm based on the Bayes rule that assumes the attributes $X_1, \ldots, X_n$ are all mutually independent, given a condition Y. The importance of this assumption lays in that it dramatically simplifies the representation of

the conditional probability P(X|Y), as well as the problem of estimating it from the training data.

Consider, for example, the case of two variables, where the feature vector $X = <X_1, X_2>$. In this case we have:

$$\begin{aligned} P(X \mid Y) &= P(X_1, X_2 \mid Y) \\ &= P(X_1 \mid X_2, Y)\, P(X_2 \mid Y) \\ &= P(X_1 \mid Y)\, P(X_2 \mid Y), \end{aligned}$$

where the second line follows from a general property of probabilities, and the third line follows from the definition of conditional independence. More generally, when the feature vector X contains *n* attributes that are mutually independent given Y, we have:

$$P(X_1,..., X_n \mid Y) = \prod_{i=1}^{n} P(X_i \mid Y) \tag{1}$$

For the sake of completeness, let us now derive the Naive Bayes algorithm, assuming in general that Y is any discrete-valued variable, and the attributes $X_1, \ldots, X_n$ are any discrete or real valued attributes. Our goal is to train a classifier that will output the probability distribution over possible values of the target Y, for each new instance X that is the data point to be classified.

The expression for the probability that Y will take on its *k*-th possible value, according to the Bayes rule, is

$$P(Y = y_k \mid X_1,..., X_n) = \frac{P(Y = y_k) P(X_1,..., X_n \mid Y = y_k)}{\sum_j P(Y = y_j) P(X_1,..., X_n \mid Y = y_j)}, \tag{2}$$

where the sum is taken over all possible values y_j of Y. Now, assuming that all X_i are conditionally independent given Y, we can rewrite the above equation as

$$P(Y = y_k \mid X_1,..., X_n) = \frac{P(Y = y_k) \prod_i P(X_i \mid Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i \mid Y = y_j)}. \tag{3}$$

This is the fundamental equation underlying the Naïve Bayes classifier (called naïve because the independent assumption if often assumed without thorough justification).

The training file comprises different lexical similarity matching scores, separated by comma. It also includes the target class of each text pair from the gold standard values. An example of the training file in the WEKA format is shown in Figure 3 (obviously, not all data rows are shown).

This file is fed into the WEKA toolset, with the Naïve Bayes option for the classification algorithm to use. The toolset automatically evaluates the average accuracy of the classification achieved on these training data.

```
@relation EVALITA
@attribute N-Gram real
@attribute Text-Similarity real
@attribute Part-of-Speech real
@attribute Named Entity real
@attribute Chunk real
@attribute class {YES,NO}
@data
24,16,10,2,15,YES
39,12,23,0,17,YES
41,15,17,1,11,YES
61,13,28,3,21,YES
78,16,34,0,9,NO
```

Fig. 3. Example of the feature vector structure of the training data

4 Experimental Results

The development and test datasets consist of 400 Text–Hypothesis pairs. The lexical features are calculated for both development and test datasets. Matching scores of the development dataset were used to train the model. The WEKA toolset was used to train the classification model and test the output for the unseen pairs thereafter.

The experimental results obtained for both development and text data predicted by the WEKA toolset using the Naïve Bayes as the classification algorithm are as follows.

The confusion matrix for the development data is shown in Table 1.

Table 1. Confusion matrix obtained on the training dataset

Correctly Classified Instances	210	52.5%
Incorrectly Classified Instances	190	47.5%
Total Number of Instances	400	

The precision, recall and the corresponding F-measure for the development dataset are shown in Table 2.

Table 2. Precision, recall, an F-measure obtained on the training dataset.

Class	Precision	Recall	F-measure
YES	0.541	0.905	0.677
NO	0.344	0.061	0.104
Weighted Avg.	0.452	0.525	0.419

The accuracy for the test dataset was 0.525.
The confusion matrix for test data is shown in Table 3.

Table 3. Confusion matrix obtained on the test dataset

Correctly Classified Instances	264	66.0 %
Incorrectly Classified Instances	136	34.0%
Total Number of Instances	400	

The precision, recall and the corresponding F-measure obtained on the test dataset are shown in Table 4.

Table 4. Precision, Recall, F-Measure on Test Data

Class	Precision	Recall	F-Measure
YES	0.602	0.945	0.735
NO	0.872	0.375	0.524
Weighted Avg.	0.737	0.660	0.630

Finally, the accuracy obtained for test dataset was 0.660. This is a very good accuracy given that no language-specific (for the Italian language) tools were used for feature extraction. Instead, all linguistic processing was performed on the English text obtained via automatic translation.

5 Conclusions and Future Work

Results show that a lexical based approach appropriately tackles the textual entailment problem. Experiments have been initiated for a semantic and syntactic based RTE task.

The next step is to carry out detailed error analysis of the present system and identify ways to overcome the errors. In the present task, the final textual entailment system has been optimized for the entailment YES/NO decision using the development set.

Finally, our textual entailment system is to be applied in Japanese, French, Italian, Spanish, and German datasets also. With those experiments we expect to show that the idea of using only English-language linguistic information for deep processing of data in other languages can be applied to a wide variety of languages, most probably depending on the quality of the automatic translation system available for this. We also plan to investigate the applicability of this idea to the cross-lingual or multilingual settings: when the hypothesis and the text are in different languages, and the training and test datasets contain pairs in different combinations of languages.

Acknowledgements. The work was partially supported by the Governments of India and Mexico under the CONACYT-DST India (CONACYT 122030) project "Answer Validation through Textual Entailment", the Government of Mexico under the CONACYT 50206-H project and SIP-IPN 20121823 project through Instituto Politécnico Nacional, and the Seventh Framework Programme of European Union, project 269180 "Web Information Quality Evaluation Initiative (WIQ-EI)".

References

1. Ledeneva, Y., Sidorov, G.: Recent Advances in Computational Linguistics. Informatica. International Journal of Computing and Informatics 34, 3–18 (2010)
2. Dagan, I., Glickman, O., Magnini, B.: The PASCAL Recognising Textual Entailment Challenge. In: Proceedings of the First PASCAL Recognizing Textual Entailment Workshop (2005)
3. Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., Szpektor, I.: The Second PASCAL Recognising Textual Entailment Challenge. In: Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy (2006)
4. Giampiccolo, D., Magnini, B., Dagan, I., Dolan, B.: The Third PASCAL Recognizing Textual Entailment Challenge. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, Czech Republic (2007)
5. Giampiccolo, D., Dang, H.T., Magnini, B., Dagan, I., Cabrio, E.: The Fourth PASCAL Recognizing Textual Entailment Challenge. In: Text Analysis Conference (TAC) 2008 Notebook Proceedings (2008)
6. Bentivogli, L., Dagan, I., Dang, H.T., Giampiccolo, D., Magnini, B.: The Fifth PASCAL Recognizing Textual Entailment Challenge. In: Proceedings of the Text Analysis Conference (TAC) 2009 Workshop. National Institute of Standards and Technology, Gaithersburg (2009)
7. Bentivogli, L., Clark, P., Dagan, I., Dang, H.T., Giampiccolo, D.: The Sixth PASCAL Recognizing Textual Entailment Challenge. In: Text Analysis Conference (TAC) 2010 Notebook Proceedings (2010)
8. Bentivogli, L., Clark, P., Dagan, I., Dang, H., Giampiccolo, D.: The Seventh PASCAL Recognizing Textual Entailment Challenge. In: Text Analysis Conference (TAC) 2011 Notebook Proceedings (2011)
9. Yuret, D., Han, A., Turgut, Z.: SemEval-2010 Task 12: Parser Evaluation using Textual Entailments. In: Proceedings of the SemEval 2010 Evaluation Exercises on Semantic Evaluation (2010)
10. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1) (2009)
11. Lappin, S., Leass, H.: An Algorithm for Pronominal Anaphora Resolution. Computational Linguistics 20(4), 535–561 (1994)