A Survey on Different Text Entailment Recognition Techniques.

Author: Mohammad Derakhshan
Supervisor: Professor Alfio Ferrara

Computer Science Department, University of Milan, Milan 20122, Italy
m.derakhshantalkhouncheh@studenti.unimi.it
alfio.ferrara@unimi.it

Abstract. Recognizing Textual Entailment (RTE) was proposed as a unified evaluation framework to compare the semantic understanding of different NLP systems[1]. in this survey, we will see some of these techniques and compare their result.At first, we explain some heuristic approaches offered by different people for solving this problem. Then, we extract features of Text and hypothesis to train machine learning models to solve the problem. In the end, we try to achieve better performance by combining different machine learning approaches. the final model has an accuracy of 88%, Precision of 84%, and Recall of 84%

Keywords: Textual Entailment · Natural Language Inference · Recognizing Textual Entailment

1 Introduction

1.1 Definition

Recognizing Textual Entailment (RTE) or Natural Language Inference (NLI) is the task of understanding whether a sentence implies the other sentence. The relation is denoted by T→H, which means concerning T (Text), we can infer H (Hypothesis). RTE has lots of applications. For example, it is used in text summarization, question answering systems, evaluating NLP models[1], Information Retrieval, Information Extraction, Machine Translation Evaluation, etc.

1.2 Datasets

For this purpose, there are different Datasets. The most famous ones are: Pascal RTE challenges dataset[2], MNLI[3], and SNLI[4]. This article used a combination of Pascal RTE datasets and SNLI to achieve 7000 samples for train and 2000 samples for test. Examples are constructed based on news and Wikipedia text in Pascal RTE datasets[5]. On the other hand, The Stanford Natural Language Inference (SNLI) is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral[6]. A drawback of SNLI corpus is that all of the sentences were

extracted from a single genre - image captions and are limited to descriptions of concrete visual scenes, rendering the sentences short and simple and making the handling of many fundamental phenomena like tense, belief, and modality irrelevant to task performance[4]. To remedy these limitations of SNLI, (Nangia et al., 2017) have released a Multi-Genre NLI (MNLI) corpus. MNLI consists of 433k sentence pairs collected similarly to SNLI. However, unlike that corpus, MNLI consists of ten distinct genres of written and spoken English, covering most of the complexity of the language[7].

1.3 Structure

The rest of this paper is organized as follows: in section 2, we prepare our dataset; then, in section 3, we explore different methods for RTE tasks. After that, in section 4, we compare the performance of each approach by considering three measurements of Accuracy, Precision, and Recall. Finally, in section 5, we conclude the paper.

2 Preprocessing

As we introduced in the introduction, there are mainly three datasets for this purpose. In this paper, we used a combination of them. for the train part, we combined RTE3 (Dev), RTE2 (Dev + Test) , RTE1 (Dev + Test) with 6,663 samples from SNLI and 10,000 MNLI samples. For the test part, we used RTE3 (Test) and 2200 samples from SNLI. Hence, 20,000 samples contain the train set and 3,000 samples test set.A snippet of the train dataset is depicted in table1.

Table 1. A snippet of Train dataset

Index	Gold Label	Sentence1	Sentence2
0	entailment	The sale was made to pay Yukos...	Baikalfinansgroup was sold to...
1	other	The sale was made to pay Yukos...	Yuganskneftegaz cost US$ 27.5...
2	other	Loraine besides participating in...	”Does A Tiger Have A Necktie”...
3	entailment	”The Extra Girl”(1923) is a story...	”The Extra Girl” was produced...

In the SNLI dataset, some records have ” − ” as Gold Label. Even though it is evident, based on the paper related to this dataset, we removed such records because they don't affect the learning procedure. The same is applied to the MNLI dataset.
Furthermore, the label (entailment - other) is not similar in all databases. For example, the RTE1 uses ”Yes” and ”No” as output. We need to make them compatible.
The final step was normalization. for this purpose, we removed all punctuation marks. Also, we tried to get the primary form of words with the help of a lemmatizer and stemmer. At first, using word_tokenizer in the $NLTK$ library, we

tokenized sentences. For each token, we converted it to lower case to eliminate differences between words like *Italy*, *italy*, *iTaLy*, etc., then the *"WordNetLemmatizer"* is applied to the token, and after that, *"PorterStemmer"*. We discovered that doing normalization can increase the accuracy by 1 or 2 percent in some models.

3 Experiment Methods

3.1 Non Machine Learning Approaches

First Method, Cosine Similarity: The very first and straightforward technique for recognizing textual entailment is using similarity. If the Text and Hypothesis have a lot in common, it may imply an entailment relationship.The strategy used for computing similarity is Cosine-Similarity. We use Bag Of Word(BOW) of Text(T) and Hypothesis(H) to feed the Cosine-Similarity function. Considering cosine-similarity, we need a threshold to separate the "entailment" and "other" label. To find out the best value for the threshold, we draw a precision-recall-accuracy chart. Regarding this chart, we realized the best threshold is 0.45. the chart is illustrated in figure 1.

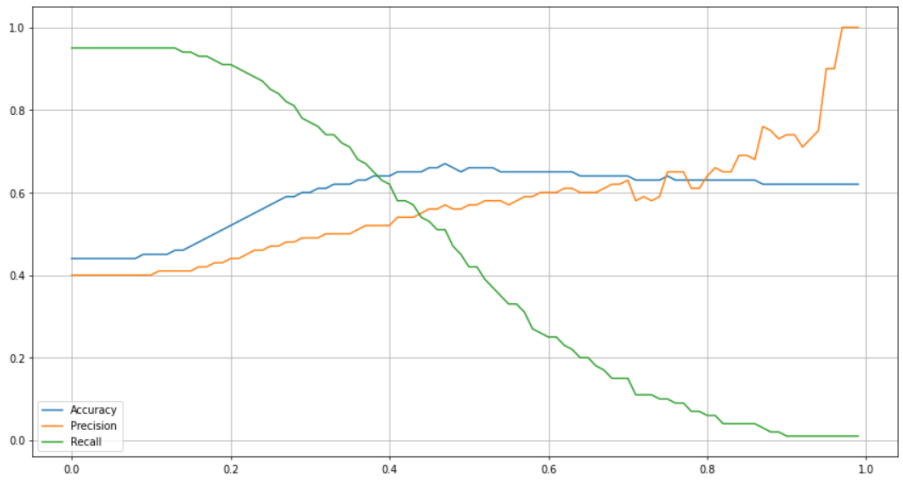

Fig. 1. Accuracy, Precision, and Recall for difference thresholds

Second Method, Cosine Directional Similarity: In 2009, D.Tatar et al[8]. introduced a directional similarity measurement for cosine similarity. They defined three cosine methods to find out whether T→ H is true or false. The

methods are defined as follows:

$$cos_T(T,H) = \sqrt{\frac{c}{m}} \tag{1}$$

$$cos_H(T,H) = \sqrt{\frac{c}{n}} \tag{2}$$

$$cos_{H\cup T}(T,H) = \sqrt{\frac{4c^2}{(n+c)(m+c)}} \tag{3}$$

Which n and m are lengths of H and T vector, respectively. also c is the number of common words of T and H.To accomplish the condition: T entails H iff H is not informative in respect to T, the similarities between T and H calculated with respect to T and to H $\cup$ T must be very closed. Analogously, the similarities between T and H calculated in respect to H and to H$\cup$T must be very closed. Also, all these three similarities must be larger than an appropriate threshold. Denoting $cos_T(T,H)$ by cos_T, $cos_H(T,H)$ by cos_H and $cos_{H\cup T}(T,H)$ by cos_{HT}, the conditions imposed are:

1. $cos_{HT} - cos_T \leq \tau_1$
2. $cos_H - cos_{HT} \leq \tau_2$
3. $max\{cos_T, cos_H, cos_{HT}\} \geq \tau_3$

The thresholds suggested by the paper are:$\tau_1 = 0.095$, $\tau_2 = 0.15$ and $\tau_3 = 0.7$. while in our experiment, we found $\tau_1 = 0.5$, $\tau_2 = 0.7$ and $\tau_3 = 0.7$ performs better.

Third Method, Modified Levenshtein Distance: The third approach is introduced by D.Tatar et al[8]. it is a modifier version of Levenshtein Distance. They defined distance as: "The minimal number of transformations (deletions, insertions, and substitutions) to transform w1 to w2".
Let's denote the distance by $LD(T,H)$. As T entails H, iff H is not informative for T the following relation must hold:

$$LD(T_{word}, H_{word}) < LD(H_{word}, T_{word}) \tag{4}$$

The paper considers the following costs for transformations:

- change case = 1,
- insert = 3 ,
- remove = 3,
- substitute = 5,
- swap = 2.

3.2 Machine Learning Approaches

We mainly concentrated on *SVM*, *Gaussian naive Bayes*, *Decision Tree*, *AdaBoost*, *MLP*, and finally, their combination.
To feed the Machine learning models, we used eight features: cosine similarity, cos_T, cos_H, cos_{HT}, $ed_{HT} - ed_{TH}$, Part Of Speed (POS) tag difference, POS tag similarity, and finally, tag similarity square.
POS difference calculates the same tag number in T and H divided by the total tags in H. POS similarity calculates cosine similarity between BOW of tags related to T and H. The models used are defined as follow:

$$SVM = SVM.SVC(kernel = rbf, C = 200, gamma = 200) \tag{5}$$

$$GNB = GaussianNB(var\_smoothing = 1e - 3) \tag{6}$$

$$DTC = DecisionTreeClassifier() \tag{7}$$

$$ADA = AdaBoostClassifier(n\_estimators = 10) \tag{8}$$

$$MLP = MLPClassifier(random\_state = 50) \tag{9}$$

Besides the above methods, we also tried to achieve a better result by combining the prediction of different machine learning models. Since the decision tree is doing well compared to the rest, we considered more weight for the prediction of this model. If the DTC recognizes an entailment relationship, we give four weights; otherwise, two weights for the prediction.

3.3 Text Expansion

After training the models, we realized the best performance is gained when we give weight to decision tree output and combine the results. The performance comparison is depicted in section 4. when we analyzed the combined model output found out this model detects 191 samples as "entailment", while they are not. (false positive) and 186 samples as "other", while they are entailment. (false negative)
So we tried the expansion technique to its impact on performance. For expansion, we used "Spacy". The definition and some examples are added to the text for each word in the sentence. For instance, if the text is " Car is Blue," the meaning of Car and Blue and some example about Car and Blue is added to the sentence "Car is Blue." the same is done for the hypothesis. In this case, the number of false-positive increased to 775, and the number of false-negative increased to 602. hence expansion harmed the result. So we removed it.

4 Performance Comparison

In this section, we compare the performance of each method regarding the approaches mentioned above. The performance is measured considering three Accuracy, Precision, and Recall parameters.

Table 2. Performance Comparision

Method	Accuracy	Precision	Recall
Cosine Similarity	0.66	0.56	0.53
Cosine Directional Similarity	0.60	0.49	0.84
Modified Lev. Distance	0.57	0.21	0.05
SVM	0.83	0.80	0.75
GaussianNB	0.66	0.54	0.64
Decision Tree	0.87	0.85	0.81
AdaBoost	0.71	0.67	0.45
MLP	0.71	0.68	0.43
Combine	0.88	0.84	0.84

5 Conclusion

As we realized in this experiment, the decision tree classifier performs well for classification purposes. Also, as we expected, SVM is the second model regarding performance. the worth approach is Modified Lev. Distance with the lowest performance. We achieved an accuracy of 88, precision of 84, and recall of 84 percent when using a combination of different models.Although we expected query expansion improves the number of true-positive, overall, the system's performance dwindled after using the expansion technique.

6 Refrences

[1] Adam Poliak: A Survey on Recognizing Textual Entailment as an NLP Evaluation.

[2] Ido Dagan, Oren Glickman, and Bernardo Magnini: The PASCAL Recognising Textual Entailment Challenge.D

[3] Adina Williams, Nikita Nangia and Samuel R. Bowman: A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference

[4] Samuel R. Bowman, Gabor Angeli, Christopher Potts and Christopher D. Manning: A large annotated corpus for learning natural language inference

[5] https://www.kaggle.com/datasets/nltkdata/rte-corpus

[6] https://nlp.stanford.edu/projects/snli

[7] https://cims.nyu.edu/ sbowman/multinli

[8] Doina Tatar, Gabriela Serban and Andreea Mihis: Textual Entailment as a Directional Relation

[9] Anish Mishra, Pushpak Bhattacharyya: Deep Learning Techniques in Textual Entailment