Combining Lexical, Syntactic, and Semantic Evidence For Textual Entailment Classification

Eugene Agichtein
Emory University
eugene@mathcs.emory.edu

Walt Askew
Emory University
wsaskew@emory.edu

Yandong Liu
Emory University
yliu49@mathcs.emory.edu

Abstract

This paper describes the Emory system for recognizing textual entailment as used for the RTE4 track at the TAC 2008 competition. We use a supervised machine learning approach to train a classifier over a variety of lexical, syntactic, and semantic metrics. We treat the output of each metric as a feature, and train a classifier on the provided data from the previous RTE tracks. As a result, our system is general, easily extensible, and naturally supports both two-way and three-way versions of the entailment task, as well as confidence estimation for the predictions. The results on both the training and the official data are promising, placing our system within the top 30% of all submissions.

1 Introduction and Overview

Our submission to the TAC 2008 RTE track relied on the analysis of shallow semantic and syntactic features. Our approach focused on designing and combining appropriate semantic and syntactic features from the annotated data set, and using the Machine Learning tools in WEKA [4] to train classifiers over theses features.

The overview of our system is shown in Figure. Our framework allows for pre-processing steps to be applied to both the text and the hypothesis, and the intermediate results stored for re-use by all the subsequent processing steps. The output of all the components for each pair are converted to corresponding feature vectors, which are in turn used to train (or test) the classifier.

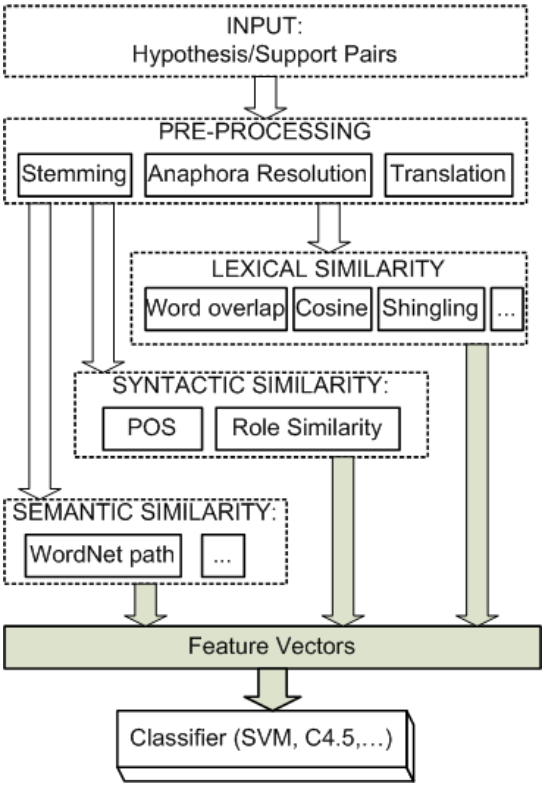

Figure 1: Overview of the Emory RTE system.

2 Features

First, the data is pre-processed with optional steps, as described next. Then, we compute three different families of similarity metrics, which could be roughly categorized as *Lexical*, *Syntactic features*, and *Semantic* metrics. The complete list of features is reported in Table 1.

2.1 Pre-processing

Stemming First, all the terms in the hypothesis and the text are stemmed using the Stanford Parser's stemmer, which is based on a finite-state transducer algorithm.

Anaphora Resolution We used the JavaRap [5] Anaphora Resolution tool over the text/ hypothesis pairs. Unfortunately, we found that JavaRap did not work as well as we hoped on the hypothesis/text pairs, perhaps due to the interactions with the stemming and other pre-processing.

Translation As an attempt to *simplify* the provided text, we experimented with translating the text to Russian, and then back to English, using Google Language Tools [3]. The intuition was that translating the text and hypothesis into foreign languages, and then back into English, might reduce the complexity of some of the sentences, making the RTE task easier.

Task The 'task' attribute provided in the XML for each text/hypothesis pair is used as a feature.

2.2 Lexical similarity

The following describes the different features we collected from the text/hypothesis pairs. Note that some of the features are created as results of sophisticated linguistic analysis.

Word Overlap The first metric we analyzed was the simplest. The word overlap metric computes how similar the text is to the hypothesis by comparing how many of the same words appear in both the text and the hypothesis. Recall that the words in the text and the hypothesis are stemmed during the pre-processing. The word overlap is then computed as below:

$$Word\ Overlap = \frac{words(text) \cup words(hypothesis)}{length(hypothesis) + length(text)} \quad (1)$$

Text Hypothesis Length Ratio The ratio between the lengths of the text and hypothesis strings (in words), computed as:

$$T\_H\_Ratio = \frac{Length\ of\ Text}{Length\ of\ Hypothesis}$$

Hypothesis Length The length of hypothesis string (in words).

Text Length The length of the text string (in words).

Cosine Similarity We tokenize and stem the text and hypothesis, then compute the cosine similarity between the term vectors.

$$cosine\_similarity = cosine\theta = \frac{A \times B}{|A||B|}$$

Substring Similarity This technique is useful because the text is usually longer than the hypothesis. First, all of the substrings of text which have the same word length as hypothesis are calculated. Then the cosine similarity is calculated between each substring and the hypothesis.

Algorithm: SubstringSimilarity
Data:
len_t = length of Text;
len_h = length of Hypothesis;
p = 0;
max = 0.0;
while *p < (len_t - len_h +1)* **do**
  S = substring(Text, p, len_h) //return substr of len_h chars from P;
  sim = the cosine similarity between Text and S;
  if *max < sim* **then**
    max = sim;
  end
end
return max;

Algorithm 1: Computing Substring Similarity between Text and Hypothesis

Median Substring Similarity This metric is similar to Substring Similarity, but returns the median cosine similarity between the substrings of text and the hypothesis.

Minimum Substring Similarity Same as Median Substring Similarity, but returns the min cosine similarity between the substrings of the text and the hypothesis.

Negative terms We detected negative terms in the text and hypothesis. The algorithm first scans through the text and hypothesis and text to detect words that imply negation such as 'not', 'no,' etc. Then a value is reported representing the existence of these negation terms in the text and hypothesis. Each value represents one of four distinct combinations of discovered negation terms. More advanced detection of negation, perhaps combined with our other features, would be more useful.

2.3 Syntatic similarity

Role Similarity Both this and the following metric use the aforementioned word overlap and WordNet similarity metrics, but on subsets of the text and hypothesis. This metric uses the Stanford parser [11] to attempt to identify the roles certain phrases play in each sentence. We use the parser to identify the phrases which map to the actor, action or object of each sentence in the text and hypothesis. The word overlap and WordNet similarity metrics are then calculated as described above between each of the discovered roles.

Separate features are calculated that measure the actor, action and object similarity between the text and hypothesis. The actor action and object are determined by traversing the parse tree returned by the Stanford parser.

POS Similarity This measure applies the word overlap and WordNet similarity metrics onto a different subset of the text and hypothesis. The Stanford Part of Speech tagger [11] is used to determine the part of speech of each word in both the text and hypothesis. Then the word overlap and WordNet similarity metrics are applied between the same parts of speech in the text and hypothesis. The set of all the nouns in the text is compared to the set of all nouns in the hypothesis, and so on for all parts of speech detected by the part of speech tagger.

The metric returns a large number of features, applying the discussed similarity metrics to determine the similarity between the set of all distinct parts of speech in the text and hypothesis.

2.4 Semantic Similarity

WordNet Similarity The next metric is somewhat more sophisticated. The similarity of the text to the hypothesis is computed using WordNet [2] to compare the relatedness of the words in the text and the hypothesis. We used the Natural Language Toolkit [9] to compute the path distance, Leacock-Chodorow Similarity [8], Wu-Palmer Similarity[12], Resnik Similarity[10], Jiang-Conrath Similarity[6], and Lin Similarity[7] between words in the text and hypothesis. As each of these metrics compare the similarity between word senses rather than words, for each word being compared, we chose its most commonly occurring word sense to calculate similarity.

Each similarity metric was averaged per text/hypothesis pair as:

$$sim = \frac{\Sigma\ Sim(words(hypothesis), words(text))}{number\ of\ similarities\ calculated} \quad (2)$$

where *sim* is one of the aforementioned WordNet similarity metrics and *Sim* is a function that returns this similarity metric between two words.

This metric calculates six different features, one for each similarity metric. Similarity could not be computed for all words in the text and hypothesis, either because some words, such as proper nouns, do not appear in WordNet, or because some of these metrics can only be calculated if an information content value has already been calculated for the word sense. Information content values represent the probability that a randomly selected word in a corpus is an instance of a given concept. These probabilities are only available for nouns and verbs in the NLTK package, and thus the only metrics available to calculate the similarity between other parts of speech using the NLTK are the path length and Wu-Palmer metrics, which do not require information content values.

3 Experiments

We used the same set of features for both the 2-way and 3-way classification tasks. First we report experimental results on the development data from the RTE tasks 1, 2, and 3. Then we summarize our official results on the RTE4 Task as provided by the organizers.

3.1 3-Way Experiments on Development Data

We first performed the entailment prediction experiment on all the possible features mentioned above, which contains 119 features and 1600 examples in all with 3-way labels, namely *YES, NO, UNKNOWN*. The accuracy is reported in Table 2.

Feature \Preprocessing	*Original*	*Anaphora*	*Translation*
Word overlap	√		√
WordNet Similarity	√		
Text Hypothesis Length Ratio	√		√
Hypothesis Length	√		√
Text Length	√		√
Cosine Similarity	√		√
Substring Similarity(Max/Min/Medium)	√		√
Task	√		
Negative terms	√		√
RoleMatching_WordOverlap	√		√
RoleMatching_WordNetSim	√		
POS_WordOverlap	√		√
POS_WordNetSim	√		

Table 1: Features and their variants used in the final submission system.

Classifier	Accuracy
Naive Bayes	0.421
J48	0.519
SMO	**0.629**
LMT	0.615

Table 2: Different classifiers on all the features with 5 fold cross-validation for 3-way task.

Since there are 119 features in all, and some of these are computationally expensive, we selected a few features which strike a balance between run time and accuracy. The features we chose to use are reported in Table 3, and their accuracy is reported in Table 4.

overlap
t_h_len_ratio
h_length
t_length
similarity
sim_shingling_min
sim_shingling_mean
sim_shingling_max
task:IE,IR,QA,SUM
trans_overlap
trans_t_length
trans_t_h_len_ratio
trans_similarity
trans_sim_shingling_min
trans_sim_shingling_max

Table 3: Selected features for 3-way task

Classifier	Accuracy
Naive Bayes	0.411
J48	0.589
SMO	**0.666**
LMT	0.654

Table 4: Different classifiers on the selected feature set with 5 fold cross-validation for 3-way task.

Almost all the classifiers actually performed much better on the smaller feature set than on the whole feature set. Thus, for our final submission, we used only the selected features.

3.2 2-Way Experiments on Development Data

We have more annotated data for the 2-way task, 2,400 text/hypothesis pairs in all. The accuracy, which was generated on the same set of selected features, is reported below in Table 5:

Classifier	Accuracy
Naive Bayes	0.573
J48	0.583
SMO	**0.596**
LMT	0.593

Table 5: different classifiers on the selected feature set with 5 fold cross-validation for 2-way task

It can be seen that our methods perform better with on the 3-way than the 2-way classification task. For some classifiers such as SMO and LMT, the performance drops significantly.

3.3 Official TAC 2008 Test Results

The results from our submission to the TAC 2008 challenge is reported in Table 6. Our best performing submission used the LMT classifier, trained by the feature set described in Table 3, and our best performance is reached in 3-way task (also judged as 3-way) with an average precision of 0.599.

3.4 Analysis of Training Results

Information Gain	Feature
0.09023	sim_shingling_max
0.09023	trans_sim_shingling_max
0.08285	parser_sim_max
0.07661	overlap
0.07115	similarity
0.07115	trans_similarity
0.06433	WordNet Similarity_jcn Similarity
0.0542	task
0.04865	sim_shingling_mean
0.04409	WordNet Similarity_path Similarity

Table 7: Top 10 features on the 3-way task with information gain

Note that there are some features with very low information gain that improved performance only slightly. These features that were deemed to have computational costs not worth the information gain were not used in our final run.

The most helpful features in this task are those that attempt to calculate similarity, as is consistently shown by analysis from previous RTE tasks. However, we proposed some variants of simple similarity computations that seem to improve performance.

Some similarity features from WordNet also have high information gain and seem useful for this task. However, they did not improve the performance in practice. One possibility is that the *sim_shingling_max* and *trans_sim_shingling_max*, which have very high information gain, are also similarity features. Thus, the gains that would come from WordNet features are subsumed by the gains from the shingling features.

4 Discussion

There are several areas which could benefit from future work.

The role matching features, although they seem intuitively valuable, were not as helpful as we had hoped. We believe this is due to the inherent complex sentence structure of the RTE examples, which are text from business news domain. Therefore, the Stanford parser which we used in this task to recognize roles phrases play in each sentence does not always give us an accurate parsing.

metric	accuracy
Word Overlap	0.513
WordNet Similarity	0.591
Role Word Overlap	0.549
Role WordNet Similarity	0.513

Table 8: results from using the LMT classifier on select features on the 3-way training data

One counter-intuitive finding from our experiments is that while the WordNet similarity metrics proved more valuable than the simpler word overlap measures when applied to the entirety of the text and hypothesis, they perform worse than the word overlap feature when role matching is attempted. One possible reason here might be that, while WordNet Similarity or Role Similarity can each provide some useful information for this task, when one is applied first such as Role Similarity, the inaccurate result of role detection can not be fully used by Word Similarity.

Another shortcoming of our approach was the inability of our classifier to reliably identify text/ hypothesis pairs as having no entailment relationship (the "No" class). This deficiency can be seen in the confusion matrix, which shows that we never predict the "No" class.

True\Predicted	Yes	Unknown	No
Yes	0.414	0.100	0
Unknown	0.167	0.209	0
No	0.862	0.144	0

Table 9: confusion matrix from all combined features

One possible explanation is that the sparsity of 'No' values in the training data (161 pairs for No, versus 617 and 822 for Unknown and Yes, respectively) resulted in classifiers that could not reliably predict no entailment. Whether this is a classifier issue, or our methodology is simply not useful for detecting 'no entailment' is not immediately clear.

Using Google language tools to translate a text as a pre-processing step is an idea that could be explored more deeply. Due to time constraints, we were only able to test this pre-processing step with a subset of

Task	Judging	accuracy	average precision
2-way	-	0.588 (top 30%)	0.600 (top 30%)
3-way	2-way	0.583 (top 30%)	-
	3-way	0.547 (top 30%)	0.599 (ranked 2nd)

Table 6: Accuracy and Average Precision on the RTE4 Test data (including approximate relative standing).

the features, and more work could be done to determine the optimal use of this approach. Translating the text from one language to the next, and then perhaps finally translating it back into English, could potentially simplify the grammatical complexity of some of the text/ hypothesis pairs and perhaps increase the effectiveness of classification.

References

[1] C.J. van Rijsbergen, S.E. Robertson and M.F. Porter, 1980. New models in probabilistic information retrieval. London: British Library. (British Library Research and Development Report, no. 5587).

[2] Christiane Fellbaum, 1998. Wordnet: An Electronic Lexical Database. Cambridge: MIT Press.

[3] Google Language Tools. http://www.google.com/language_tools.

[4] Ian H. Witten and Eibe Frank, 2005. "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005. Machine Learning, 34(1-3):43-69.

[5] JavaRAP. http://www.comp.nus.edu.sg/ qiul /NLPTools/JavaRAP.html

[6] Jiang J. and Conrath D. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of International Conference on Research in Computational Linguistics. Taiwan.

[7] Lin D. 1998. An information-theoretic definition of similarity. In Proceedings of the 15th International Conference on Machine Learning. Madison, WI.

[8] Leacock C. and Chodorow M. 1998. Combining local context and WordNet similarity for word sense identification. In Fellbaum 1998, pp. 265-283.

[9] Natural Language Tool Kit. http://nltk.sourceforge.net

[10] Resnik P. 1995. Using information content to evaluate semantic similarity. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, pages 448-453. Montreal.

[11] Stanford Parser. http://nlp.stanford.edusoftware/lex-parser.shtml

[12] Wu Z. and Palmer M. 1994. Verb Semantics and Lexical Selection. In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics. Las Cruces, New Mexico.