Textual Entailment Using Univariate Density Model and Maximizing Discriminant Function

Scott Settembre
SNePS Research Group, Department of Computer Science and Engineering
University at Buffalo
Buffalo, NY 14214, USA
ss424@cse.buffalo.edu

Abstract

The primary focuses of this entry this year was firstly, to develop a framework to allow multiple researchers from our group to easily contribute metrics measuring textual entailment, and secondly, to provide a baseline which we could use in our tools to evaluate and compare new metrics. A development environment tool was created to quickly allow for testing of various metrics and to easily randomize the development and test sets. For each test, this RTE tool calculated two sets of results by applying the metrics to both a univariate Gaussian density and by maximizing a linear discriminant function. The metrics used for the submission were a lexical similarity metric and a lexical similarity metric using synonym and antonym replacement. The two submissions for RTE 2007 scored an accuracy of 61.00% and 62.62%.

1 Introduction

The task of textual entailment for the PASCAL Textual Entailment Challenge for 2007 was to create a system to determine if a given pair of sentences, called the Text-Hypothesis (T-H) pair, had the property of having the Text sentence entail the Hypothesis sentence. Each Text-Hypothesis pair is also assigned the type of entailment that should be applied to the pair when evaluating its entailment. There are four types of entailment, each of which may or may not need different techniques to determine entailment, and for the purposes of the RTE tool developed, are considered separate problems.

2 RTE Development Environment Tool

Our research group decided to begin focusing on the Recognizing Textual Entailment challenge this year in February and to continue our participation for years to come. It was decided to create a development environment from which our researchers could attempt different techniques of examining a Text-Hypothesis pair and yet all metrics resulting from those techniques could be used in calculating the final results. The RTE tool also randomly generates training and testing sets from the 800 Text-Hypothesis pairs provided for development by the competition to avoid over-fitting the data during the training stage.

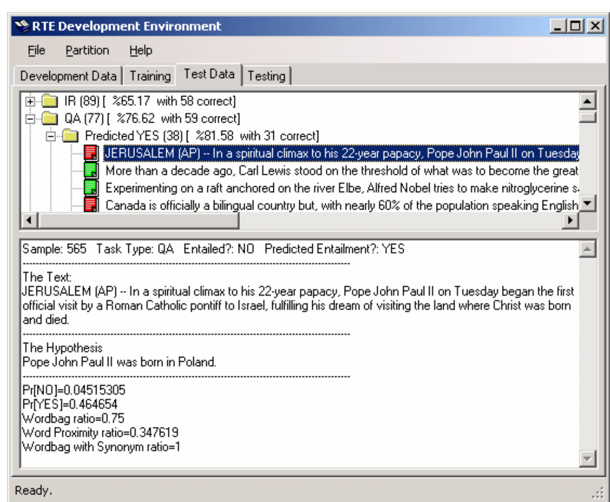

Figure 1. Screenshot of the RTE Development Environment.

Proceedings of the Workshop on Textual Entailment and Paraphrasing, pages 95–100,
Prague, June 2007.

The RTE Tool can generate a metric by calling a .NET object, COM object, web page, command line, or an internal function. These metrics are cached to speed testing, though a specific metric type can be cleared manually should the object or function generating the metric be changed.

In the image of the RTE tool above, we can see a typical results screen. We have a misclassified sample highlighted and all the relevant data for that sample displayed on the bottom. Each category is represented with a folder and displays the accuracy results of the last classification. In this way, we can easily compare and contrast different metrics and their effectiveness on the samples in a simple and intuitive way.

2.1 Defining Metrics

Each metric developed is required to produce a continuous variable that can measure a feature of the T-H pair. The metric value is required to be normalized between 0 and 1 inclusive so that we can use the same metrics for future expansion when possibly dealing with nearest-neighbor classification techniques and not be subject to scaling issues. This is also valuable if we intend to develop vague predicates [Brachman and Levesque, 2004] to use in Boolean rules, another potential classification implementation.

There is also currently a constraint that the metric value "0" means the least entailment (according to that particular metric) and the value "1" means the most entailment. This helped create an easy way to maximize our linear discriminant function (which will be described below). This constraint is unnecessary when classifying using the univariate density model.

2.2 Classification Methods

The tool classifies a T-H test pair using one of two classification methods. The first method uses the metrics of the training set to generate the parameters for eight Gaussian distributions, or two distributions for each type of textual entailment. Each distribution describes a *probability density function* (PDF) for a particular type of entailment. For example, there is one PDF for the entailment type of "Question Answering" (QA) whose entailment is "YES", and there is one PDF for the entailment type of QA whose entailment is "NO". This univariate normal model was chosen to simplify the calculations over the multivariate model we planned to use. Since the submissions would only consider one metric for each run, instead of using all the metrics we have defined, the univariate model was appropriate.

The second method of classification uses the metrics from the training set to develop a linear decision boundary to maximize the accuracy outcome in the test set. Once this boundary, or threshold, is determined for each of the four types of entailment, a simple comparison of the metric from a T-H pair can be classified based on what side of the boundary it is on. This linear discriminant function had a further constraint that required the metric values be described in a certain way to simplify the classification function. This requirement will be lifted for our next submission in order to deal with solution landscapes that may not adhere to our Gaussian distribution model.

3 Metric Set Used for Submission

Three different metrics were developed for use in our RTE tool this year. We decided to concentrate on producing simple measurements to create a baseline for which to judge the development of new metrics as well as to judge the performance of future training or classification methods.

Due to time constraints, we chose to employ simple metrics, which have been used before, in order to meet our primary goals. Despite the simplicity and the lack of semantic interpretation of the metrics, these metrics coupled with our pattern classification strategy yielded competitive results.

3.1 Lexical Similarity Ratio Metric

Our first metric is a simple lexical similarity ratio between the words in the Text and Hypothesis sentences in a T-H pair. The formula counts number of matches between the occurrences of a word in the Hypothesis and the words in the Text. The sum is then normalized by dividing it by the number of words in the Hypothesis itself. For baseline purposes, every word was considered and only punctuation was removed. This technique was also used by other teams in previous challenge submissions [Jijkoun and Rijke, 2005].

3.2 Average Matched Word Displacement

Our second metric was not used in the final results, but will be described for completeness. This metric was the average of the distances in the Text between matched words from the Hypothesis normalized by dividing that average by the maximum possible distance. In other words, if two words in the Hypothesis were found in the Text, the distance between them in the Text would be averaged with all the other combinations of matched word pair distances and then normalized by dividing the maximum possible distance value for that particular sentence. Preliminary results showed a less than significant correlation and so were not used in this submission.

3.3 Synonym and Antonym Replacement

The third metric is nearly identical to the lexical similarity metric defined above except that if a word in the Hypothesis sentence is not matched, then all its synonyms and antonyms are also searched for in the Text sentence. Any synonym matches raise the score and any antonym matches lower the score by a fixed amount, and in this case arbitrarily selected as ±1 (before normalization). A Microsoft Word 2003 COM object was used to search for the synonyms and antonyms from Microsoft Word's lexical database.

4 Classification used for Submission

Two different types of classification methods were used to classify entailment for a Text-Hypothesis pair. Both types are described below.

We chose to initially keep our classification models simple and easy to visualize so that both our experienced and inexperienced research group members could participate. The "No Free Lunch Theorem" [Duda, Hart, and Stork, 2001] shows that there is no inherent benefit to any specific classifier[1], and since the more important task of generating the metrics[2] crosses academic disciplines in our research group, we found communicating in terms of a Gaussian distribution was easily understood.

[1] For "good generalization performance, there are no context-independent or usage-independent reasons to favor one learning or classification method over another."

[2] Since we are creating the metrics, we are attempting to distribute the values in a Gaussian curve. This becomes a "context" which we can favor a classifier that will classify the data better, such as the univariate normal model. Our goal is to create a better metric and not necessarily to find a better classifier.

4.1 Univariate Normal Model

The continuous univariate normal model, or Gaussian density, allows us to calculate p(x), or the probability that feature x will appear in a dataset. The data points in the given dataset is assumed to be distributed in a Gaussian distribution, sometimes referred to as a bell curve. Of course if the data points in that data set turn out to be distributed in a non-Gaussian curve (i.e. exponential curve or even linear) or multimodal curve (more than one peak), then we may not be able to draw any conclusions. For the purposes of our metrics, we are assuming a Gaussian distribution, and encourage the developer of the metric function to attempt to fit the metric results into Gaussian curve.

The two parameters of interest are the mean $\boldsymbol{\mu}$ and the variance $\boldsymbol{\sigma^2}$, of the data points. With these two parameters, we are essentially able to calculate the *probability density function* (PDF) for the category. After calculating these parameters from the development data set, we can apply the following formula to generate the probability, *p*(x), of a sample, where x is the metric value we wish to classify.

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

During the training step, the mean of a category is calculated. The following formula does this calculation, where n is the number of samples, and x_i is a particular metric of the i^{th} sample:

$$\mu = \frac{\sum_{i=1}^{n} x_i}{n}$$

Also during the training step, the variance of a category is also calculated, with this formula:

$$\sigma^2 = \frac{\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2 / n}{n}$$

For each type of entailment, there are two classifiers: one classifier for "YES" and one classifier for "NO", representing the two categories. During the *training* step, the mean and variance parameters are calculated directly from the metrics that come from the development data. During the *testing* step, the specified metric is calculated for the T-H pair, and using the univariate normal formula, we can calculate the probability that the calculated metric is in the "YES" category or the "NO" category. Then which ever result is larger, that category is chosen as the answer.

To understand the limitations of this method, we have a quick example. Here is a parameter list of each category as well as the decisions that were made from them:

(IE,NO) = { μ = 0.6867668 , σ = 0.1824087}
(IE,YES) = { μ = 0.6874263 , σ = 0.1622136}
(IR,NO) = { μ = 0.3649016 , σ = 0.1984567}
(IR,YES) = { μ = 0.5888839 , σ = 0.2035728}
(QA,NO) = { μ = 0.4470804 , σ = 0.1821738}
(QA,YES) = { μ = 0.7330091 , σ = 0.1873602}
(SUM,NO) = { μ = 0.4470848 , σ = 0.2625011}
(SUM,YES) = { μ = 0.657442 , σ = 0.250246}

Overall correct entailments made: 492 out of 800.
Overall probability of success : 0.615

IE (200) [%47.5 with 95 correct]
  Predicted YES (0) [%NaN with 0 correct]
  Predicted NO (200) [%47.5 with 95 correct]
IR (200) [%66.5 with 133 correct]
  Predicted YES (76) [%63.16 with 48 correct]
  Predicted NO (124) [%68.55 with 85 correct]
QA (200) [%73.5 with 147 correct]
  Predicted YES (95) [%77.89 with 74 correct]
  Predicted NO (105) [%69.52 with 73 correct]
SUM (200) [%58.5 with 117 correct]
  Predicted YES (133) [%60.9 with 81 correct]
  Predicted NO (67) [%53.73 with 36 correct]

As we can see, the two categories (IE,NO) and (IE,YES) are very similar in mean, μ. This essentially translates to two Gaussian curves peaking at the same point, which would cause an overlap that would favor the curve with the larger variance during the calculation of p(x). If we look at the results using these parameters, we can see that in the "IE" type of entailment all decisions were made in favor of that category. This does not mean that there is an error, just that the distribution of this metric is too similar and so probably is not a good metric to use in deciding the classification for that category. Whereas in entailment type "QA", we find that this metric does indeed divide the categories into two curves that are quite separated, and so yields a good accuracy.

4.2 Maximizing the Discriminant Function

This is the easiest way the RTE tool calculates whether a T-H pair is in a specific category. If a metric is less-than a specific threshold, then the T-H pair is classified as "NO", and if it is above the threshold, then the pair is classified as "YES". Each type of entailment has its own discriminant function and therefore, there are only four classifiers or in this case, technically defined as four *dichotomizers*.

Each threshold is calculated using a brute force iterative technique. After the metric is calculated for each sample, the RTE tool simply increments the threshold a certain fixed amount (arbitrarily selected as 0.001 each each iteration) and records the accuracy over the entire development data set for that iteration. As the process concludes after one thousand iterations (that is, moving the threshold from 0 to 1 in .001 increments), the threshold with the maximum accuracy is selected as the threshold for that classifier. This, however, places a constraint on the way the metric needs to be defined, as described above in section 2.1.

5 Results

There are four result sets representing each of the metrics used paired with each of the classification strategies used. The first table below shows the actual results, broken down into each type of entailment, using the released annotated test set. The second table shows our results by randomly splitting the development dataset 80%/20% into a training set (80%) and a testing set (20%). From the results listed in the second table, it was decided which metric/classification pair would be used in our final submission.

Although we cannot truly compare results from this competition to last years RTE 2 competition, we found that our results seemed quite competitive. [Bar-Haim, Dagan, et al. 2006] We do recognize that some of our metrics have already been employed by other teams [Jijkoun and Rijke, 2005] and that our results may be different because of the thesaurus corpus we employed and the classification strategy we used.

5.1 Actual Results

The actual results are based on training the RTE tool we developed on the released annotated development dataset and then applying the trained classifiers on the test dataset. In this table, each column represents a training metric used with a classification method. For the two metrics used, "LS" represents Lexical Similarity, while "LR" represents Lexical Similarity with Synonym and Antonym Replacement (or Lexical Replacement for short). For the two types of classification used, "UN" represents the Univariate Normal model, while "DM" represents Linear Discriminant Maximization.

	LS+UN	LR+UN	LS+DM	LR+DM
Overall	0.615	**0.626**	**0.61**	0.629
IE	0.475	0.510	0.495	0.505
IR	0.665	0.630	0.635	0.640
QA	0.735	0.750	0.750	0.750
SUM	0.585	0.615	0.560	0.620

As the reader can see, our final submissions' scores were not the maximal ones from the table. Our first submission we submitted scored 61% and our second submission scored 62.62%. For our first submission, the Lexical Similarity metric was used in conjunction with the Linear Discriminant Maximization model for classification. For our second submission, our Lexical Replacement metric was used in combination with the Univariate Normal model of classification. These two combinations were chosen, however, from the training results below.

5.2 Training results

Using these results, it was decided to pick the maximal overall accuracy using both metrics. It was assumed that the same correlations found in the development dataset would be found in the testing dataset. Though this did not ring true in actuality, the final results using either method were quite close.

	LS+UN	LR+UN	LS+DM	LR+DM
Overall	0.669	**0.675**	**0.717**	0.644
IE	0.425	0.575	0.625	0.600
IR	0.688	0.667	0.688	0.646
QA	0.811	0.784	0.811	0.784
SUM	0.771	0.686	0.775	0.543

6 Conclusions and Future Enhancements

The lexical similarity metric and its variants obviously have some correlation to whether a Text-Hypothesis pair has entailment or not. Though we were surprised by the results (from our runs exceeding results from other teams' runs from previous years) and at how well they worked initially, further investigation found the accuracy of certain types of entailment, especially Information Extraction (IE), lacking and perhaps making some metrics almost irrelevant as a viable metric.

By focusing our efforts this year on developing a tool to test various methods of classification and metrics, we created an excellent way to develop our ideas and distribute our research efforts among researchers. The RTE Development Environment will help us coordinate our efforts and allow small gains in any individual metric to contribute to the overall classification in a proportionately significant way.

For future enhancements, we intend to apply the multivariate model to process a metric vector in determining classification instead of just considering one metric at a time (as we did in the univariate model). In addition, we intend to extend our metrics to consider semantic interpretations and comparisons between the Text-Hypothesis pair.

We feel that our overall success was illuminating to the larger task at hand and we are looking forward to applying our decision making framework to next year's submission. Judging by our results, the simplicity of our approach will quite possibly yield a competitive entailment strategy even in comparison to more syntactic or semantic decompositions of the sentence pairs at this time.

Our primary success, over the three week period in which we addressed this problem, was the development of a tool and a process by which members of our research group can interact. The pooling of expertise from our linguistics, computer science, and cognitive science disciplines and constructing our future plan of action culminated in the development of this tool, benchmarks for our group, and constraints in which we can operate efficiently and address this problem with more depth in the future.

7 Acknowledgements

We would like to thank Dr. Stuart Shapiro and Dr. William Rapaport of the SNePS Research Group, University at Buffalo, for their encouragement and guidance in this year and in the years to come.

Special thanks to Dr. Sargur Srihari of CEDAR, "Center of Excellence for Document Analysis and Recognition", University at Buffalo, for providing insight into various classification techniques. Finally, we congratulate our members of the SNePS Research Group for their contributions over the short amount of time we had to address this challenge this year.

References

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini and Idan Szpektor. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Ronald J. Brachman and Hector J. Levesque. 2004. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers, San Francisco, CA.

Richard O. Duda, Peter E. Hart, David G. Stork. *Pattern Classification*. Wily, New York, second edition, 2001.

Valentin Jijkoun and Maarten de Rijke. *Recognizing Textual Entailment Using Lexical Similarity.* Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment, 2005.