Semantic Coherence

Elena Eneva, Rose Hoberman, Lucian Vlad Lita
{eneva,roseh,llita}@cs.cmu.edu

November 30, 2000

1 Introduction

1.1 Motivation

Perhaps the most salient deficiency of conventional language models is their complete failure at modeling semantic coherence. These models capture fairly well short distance correlations among words in a sentence, yet are unable to distinguish meaningful sentences where the content words come from the same semantic domain from 'fake' sentences where content words are drawn randomly. As a result, in many language technology applications such as speech recognition, errors that are obvious to a human observer (e.g. a noun replaced by an acoustically similar but semantically different noun) cannot be salvaged by the model. For example, if the speaker says *Is there a plane at four from the Pittsburgh airport*, and the system recognizes *arrow port* instead of *airport,* we would like to automatically identify and correct this error.

The whole-sentence exponential language model developed recently by Roni Rosenfeld's group is naturally suited to modeling whole-sentence phenomena such as semantic coherence. To incorporate semantic coherence into this exponential model, we need to derive a measure of whole sentence coherence. This coherence measure could be used to improve the original language model, and thus improve performance in any application where language modeling is useful.

1.2 Project Description

In order to determine a good semantic coherence measure we built a classifier which distinguishes between real sentences generated by a human and fake sentences generated by a conventional language model. The confidence score produced by the classifier is used as an indicator of semantic coherence.

1.3 Previous Work

[Rosenfeld] introduced a maximum entropy model which directly models the probability of an entire sentence. In later work, [Zhu] used the maximum entropy model with linguistic features that captured variable length syntactic sequences and achieved a slight improvement of the original model.

[Cai] explored a complementary aspect of language, namely within-sentence semantic coherence. A small set of coherence statistics were extracted from each sentence and added as features to the exponential model. These features improved the model by 3 to 5%, but were not used directly to classify sentences based on coherence.

[Dagan] has addressed the issue of how to estimate the likelihood of co-occurences that do not occur in the training data. Their method makes local analogies between each specific unobserved co-occurence and other co-occurences that contain similar words.

[Cox]has used semantic information to estimate the confidence that individual words output by a speech recognizer are correct. They used latent semantic analysis to get a measure of semantic similarity between word pairs, and then used the average semantic similarity of a word to all the other words in the sentence as their confidence that the word was correct. Their method hurt the performance of the N-best list when the number of words considered was high, but out-performed the N-best list when the number of words considered was low. They did not try their method for classifying the entire sentence as coherent or not.

1.4 Results

The best classification accuracy our learning algorithms have achieved is about 80% using all the features and a standard stoplist. This is a significant performance improvement over the results obtained by using features developed in previous work.

2 Approach

2.1 Semantic Association between Word-Pairs

This project builds on previous work by [Cai] and uses the same correlation measure. For each word-pair in that dataset, we estimate a measure of association called Q (Yule's[1] statistic) based on the appropriate 2×2 contingency table of training data co-occurrences of these two words. For two words, the contingency table is:

[1]  George Udny Yule (1871-1951)

		Word 1 Yes	Word 1 No
Word 2	Yes	C_{11}	C_{12}
Word 2	No	C_{21}	C_{22}

where C_{11} is the count of sentences in the training corpus which contain both words, C_{12} is obtained by subtracting C_{11} from the number of sentences with *Word 1* in it; C_{21} is obtained by subtracting C_{11} from the number of sentences with *Word 2* in it; C_{22} is the total number of sentences minus the other three counts. Based on such tables we can compute the Q statistic for each pair of words:

$$Q = \frac{C_{11} \cdot C_{22} - C_{12} \cdot C_{21}}{C_{11} \cdot C_{22} + C_{12} \cdot C_{21}} \tag{1}$$

The values of Q range from -1 to 1; the higher the Q value, the stronger the correlation between the two words. Q is -1 when two words have never occurred in the same sentence, and is 1 when they always occur together. Each sentence, true or fake, can be represented by a variable length list of Q values–a Q value for each word pair. By defining some statistics on these Q lists and comparing the true and fake data, we observe differences in the distributions of the statistics. A classifier can use these statistics to distinguish between coherent and non-coherent sentences.

2.2 Datasets

Our dataset is comprised of approximately 250,000 sentences. Half of these sentences are from a broadcast news domain, and the other half are generated by a traditional language model (a modified Kneser-Ney trigram model). We divided this dataset into equal size training and testing sets. The correlation values are computed from another 103 million words of broadcast news data. This is the same corpus on which the trigram model was trained.

Since the trigram model is good at generating local coherence, it is incorrect to assume that short sentences produced by the model are not coherent. For example, the sentences *Mike Stevens says it's not real* and *We've been hearing about it* were generated by the trigram model, but obviously they are coherent. For this reason, we filtered out all sentences with fewer than seven words or fewer than four content words.

Our vocabulary consisted of all the words present in the 103 million word corpus (approximately 200,000 distinct words). Many of these words are function words and do not carry a semantic meaning (e.g.. *to, the, because*). Language researchers generally use a stop list consisting of function words in order to obtain better performance. We experimented with different stop lists of varying sizes. We generated five versions of the datasets filtered with stop lists of sizes ranging from 0 to 1000 function words.

2.3 Learning Algorithms

In order to train a classifier to discern between coherent and non-coherent sentences, we first generated a large set of features. We did not explore features which used syntactic knowledge, information about specific words, or lexical resources like thesauri. The features are based only on the correlation values, the co-occurence counts, and the number of content and stop words. We then used these features to train a decision tree and an ANN to classify sentences.

The decision tree algorithm used was C5.0, and the selection criterion applied was gain ratio. To avoid overfitting we pruned with a severity of 75 on a scale from 1 to 100. We also set a minimum number of training examples per rule, generally around twenty, but also experimented with other values. In addition, we used boosting on decision trees.

The ANN used was the neural network provided by Clementine, a data mining software package. The network had one hidden layer, and the number of hidden nodes was varied to find an optimal structure. We held out 50% of the training data to use for cross-validation to prevent overfitting.

We also used Rainbow, a program that performs statistical text classification. It is based on Bow, a library of C code useful for writing statistical text analysis, language modeling and information retrieval programs. Its library and front-ends were designed and written by Andrew McCallum. Using Rainbow, we built four language models (unigram, bigram, trigram, and fourgram) for the coherent and noncoherent datasets and then applying them to the testing corpus for classification purposes.

2.4 Features

We have implemented roughly eighty features, which fall into eleven classes. The majority of the feature classes rely on word pair correlation values in sentences. The feature classes are:

1. High/low correlation across large/small distances - a sentence contains one or more pairs of words which are very far apart and which have a high positive correlation. Different features can be obtained by varying the distance and correlation thresholds.

2. Simple statistics - the simple correlation characteristics of a sentence: mean, median, maximum, minimum, range and variance of word pair correlation values.

   (a) Statistics considering all word pairs

   (b) Statistics considering only pairs at least 5 words apart

3. Word clustering - identify topics by clustering words in a sentence, where the distance measure is the word pair correlation

4. Word and phrase repetition[2] - in real sentences topic words are often repeated as are some phrases

5. Percentage of high/low correlation values

6. Internal variation - a sentence has internal variation if different parts of the sentence exhibit very different statistics.

7. Unseen pairs - statistics of pairs which were never seen in the same sentence in the large training corpus

8. Content and Stop Words - percentage and longest sequence of content and stop words

9. Sentence Length

10. Product of word pair probabilities - the product of the probability of each word pair in the sentence, normalized for sentence length. The probability of a word pair is $\frac{C_{11}}{\#sentences}$. We used $+1$ smoothing so $C_{11}$was never zero.

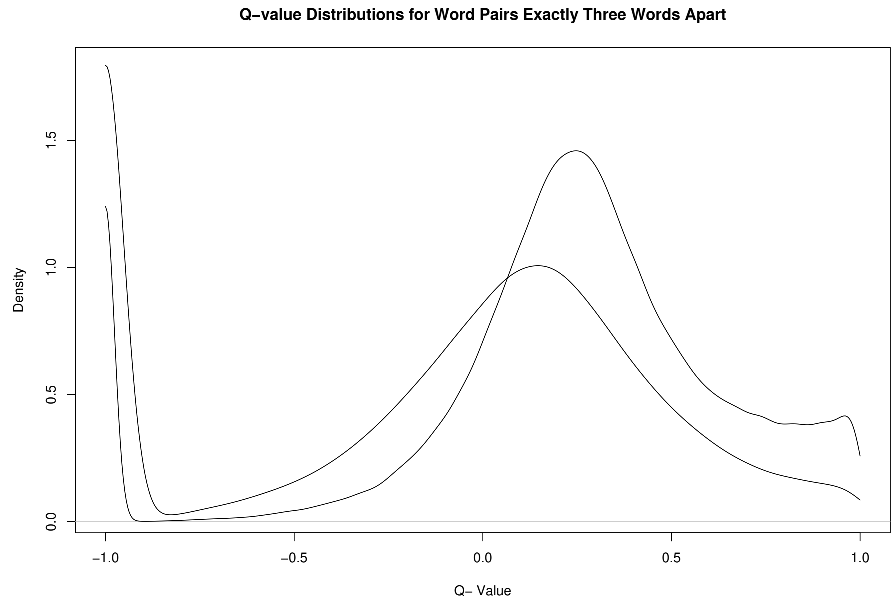
11. Likelihood Ratio - the likelihood the sentence came from broadcast news divided by the likelihood the sentence was trigram-generated

2.5 Likelihood Ratio

Instead of all of the above features, we determined that we can classify the sentences with just slightly lower accuracy using only one feature. This feature is the likelihood that the sentence came from broadcast news divided by the likelihood that the sentence was generated by a trigram language model. In a trigram language model the semantic correlation between pairs of content words is primarily determined by the distance between the two words. Thus, adjacent words should be semantically similar. As the words get farther apart, however, the effects of the trigram weakens, and the association diminishes. In real sentences, on the other hand, the semantic correlation between word pairs decreases more slowly with distance.

For each corpora we generate density estimates of the Q values conditioned on the distance between the words. Kernel density estimation is used to smooth the density estimates. The figure below shows the distributions of the Q values in broadcast news and in trigram-generated sentences conditioned on a distance of three (adjacent words are considered to be at a distance one). We selected distance three as an example because this is the smallest distance which is not within trigram distance. Notice that the broadcast sentences are twice as likely to contain words three words apart with a correlation value of 0.75 sentences than the trigram-generated sentences. But at a correlation of -0.25 the trigram model is twice as likely. Also note that the medians differ by a value of about 0.10.

[2]people stutter, machines don't



These density estimates can then be used to classify new sentences. A sentence is represented as a set of correlation and distance pairs. For each correlation we calculate the likelihood that it came from each corpus. The likelihood that a sentence came from that corpus is then just the product of each of the individual likelihoods. The ratio of the two likelihoods is a good indicator of the semantic coherence of the sentence:

$$\prod_{(Q,distance)} \frac{Density(Q|distance, broadcast)}{Density(Q|distance, trigram)} \tag{2}$$

If the ratio is a very large positive number, we are very confident the sentence is semantically coherent. If the ratio is a very large negative number, the sentence is most likely not semantically coherent. If the ratio is near one, we are uncertain to which class the sentence belongs.

3 Results and Analysis of Features

The classification accuracy for some of the experiments is shown in the following table:

Feature Set	Learner	Stop List Size	Classification Accuracy
Training Sentences - Unigram	Rainbow	500	50.98
Training Sentences - Bigram	Rainbow	500	51.15
Training Sentences - Trigram	Rainbow	500	51.09
Training Sentences - Fourgram	Rainbow	500	51.48
80 Features from Classes 1-10	C5.0	1000	50.36
Features Used in Previous Work	C5.0	50	71.47
80 Features from Classes 1-10	C5.0	0	73.25
80 Features from Classes 1-10	ANN	0	73.68
Likelihood Ratio	-	500	75.07
80 Features from Classes 1-10	C5.0	50	75.57
80 Features from Classes 1-10	ANN	50	76.65
80 Features from Classes 1-10	C5.0	500	77.71
80 Features from Classes 1-10	ANN	500	76.48
81 Features from Classes 1 -11	C5.0	500	78.53
80 Features from Classes 1-10	C5.0	630	78.65
81 Features from Classes 1-11	Boosting w/ C5.0	500	79.63

The best classification accuracy that we have obtained with different sets of features, stop lists, and learners is about 80%. This was achieved by using the C5.0 decision tree with a stop list of 630 words on 80 features belonging to ten feature classes. On all the experiments, the decision tree and the ANN gave comparable results. This is a significant performance improvement over the results obtained by using features developed in previous work.

The language models trained by Rainbow failed to distinguish coherent from noncoherent sentences. This shows that locally there is no difference between the sentences, and to derive a coherence measure it is necessary to look at global sentence statistics.

The features used in previous research were never evaluated on the classification task. In order to compare our results, we ran an experiment using a similar set of features, and obtained an accuracy of 71.47 percent. However, the original work used a much smaller vocabulary and calculated the Q values in a different way, so these results do not necessarily reflect the classification performance that would have been previously obtained.

Increasing the minimum number of training examples per rule for the decision tree improved results slightly, but it is unclear whether this improvement is statistically significant.

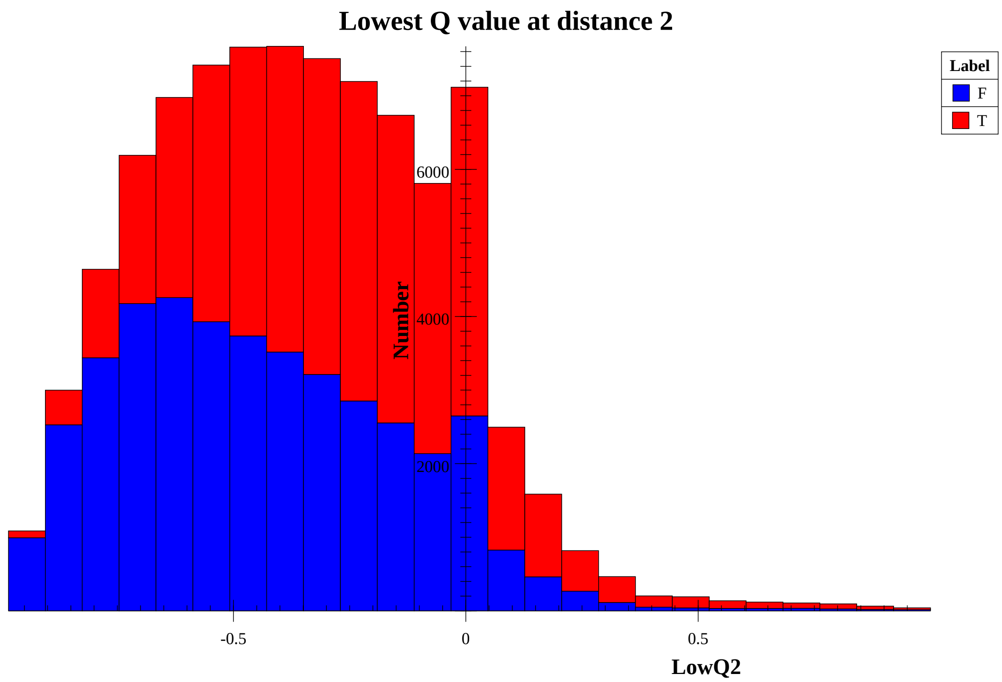
One trend that can be seen in the results is that increasing the size of the stop list tends to improve accuracy. However, when the stop list size is increased, more of the sentences have fewer than four content words and thus get filtered out. Some of the improvement in accuracy could be because of this decrease in the size of the test set. To determine if adding any words to the stop list would increase results, we created an even larger (1000 word) stop list which included 300 additional words that were borderline cases: they could be either function words or content words. In this case, however, the performance dropped down to almost the same as chance (50.36 percent). So clearly, merely adding stop words and reducing the test size does not improve results.

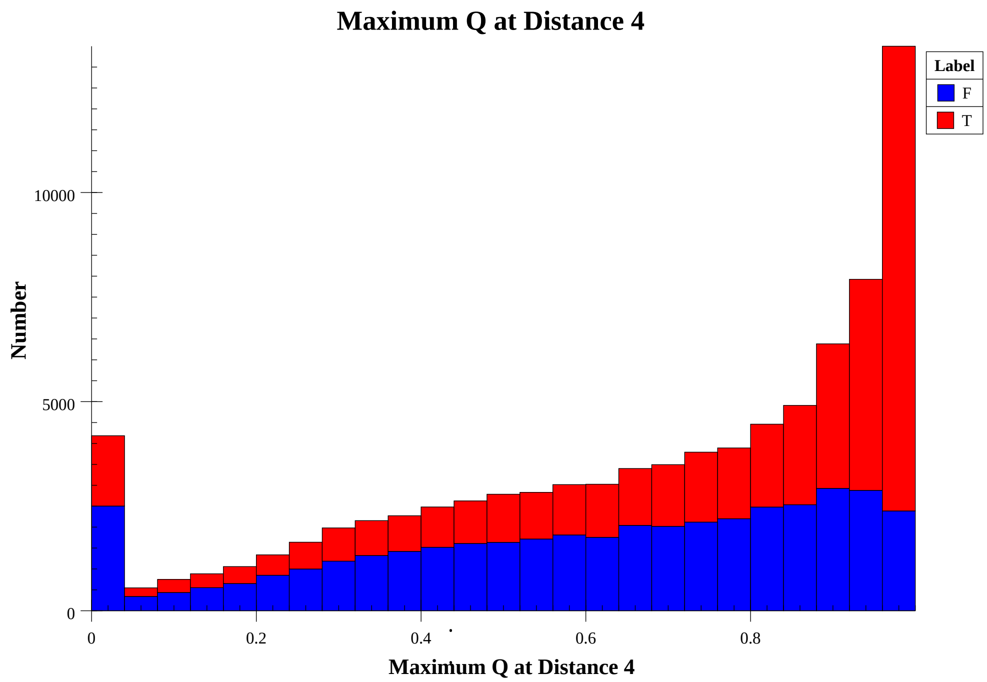
Using only the likelihood ratio feature, we obtain 75.07 percent accuracy. This accuracy is obtained if we set the threshold for "true" to be 0.93. As expected, the best threshold is close to 1, which is the value of the feature when both likelihood estimates are identical. Although this feature gives very good performance on its own, when adding this feature to the larger set of features the improvement in accuracy is only 3%. This is most likely because of redundancy in the information provided by the larger set of features.

We also ran boosting on the decision trees, and obtained higher accuracy than with just C5.0 or the neural network. However, these results were obtained using only 10 rounds of boosting. More rounds will most likely results in additional improvements.

To evaluate the usefulness of each of the features, we can look at the rules learned by the decision tree, and at the relative importance the ANN output for each feature.

Below we plot the histograms of two of our most useful features:





Notice that for the first feature, if the lowest Q value at distance two is very small then te sentence is unlikely to be from broadcast news. If the maximum Q at distance four is greater than 0.95, on the other hand, then the sentence is not likely to have been generated by the trigram model.

The features to which the ANN gave the most importance show that it is clearly learning the features that best distinguish a trigram-generated sentence from real sentences from broadcast news. For example, we used a set of features consisting of the maximum Q value at various distances. The most useful of this set was the maximum Q value at distance four. This is the distance which is outside of trigram effects but still small enough to see strong correlations in true sentences. Similarly, for the minimum Q values, the learner selected distance two as a very useful feature. This feature helps identify trigrams which are not semantically coherent. The features which looked for repetitions of words and phrases also performed well, as did the number of clusters in the sentence.

An example of a useful rule learned by C5.0 is:

Likelihood_Ratio > 0.93 *&*

Maximum_Q_at_Distance_4 > 0.915 *&*

Number_Clusters $\leq 14 \Longrightarrow$ *Coherent_Sentence*

This rule has a test accuracy of 93% on the 15,000 sentences to which it applies.

4 Future Work

The most obvious problem with using Q values as our word pair correlations is that for words that appear infrequently in our corpus the point estimates of the Q values are often unreliable. One idea for reducing the effect of these unreliable estimates is to use a confidence distribution on the Q's instead of the erroneous point estimate. Another solution to this problem would be to use a different correlation statistic. One possibility is the χ statistic. Another fourth possibility is to use the method of [Dagan] to improve the estimates of the likelihood of co-occurrences that are rare in the training data.

Since our results show that the classification accuracy is significantly affected by the size of the stop list, it might be worth running some sort of feature selection process to determine which words should be removed as stop words, and which should remain as content words.

When people are given this classification task, they tend to identify words in the sentence that belong to the same topic. We would like to try to approximate this process by using other clustering methods to group the words in the sentence. Then statistics based on these clusters could be used as additional features.

Since sentences from broadcast news tend to come from specific categories, e.g. sports, entertainment, business, it is possible that the sentences could be clustered into such categories. Then if a sentence spans multiple categories it is not likely to be semantically coherent.

Many of the features we selected perform well on a common subset of the sentences. However, some sentences are consistently misclassified. Boosting is useful for learning rules which can classify well these hard cases, and should be further explored. Support vector machines are also another possibility for a more accurate learner.

Clearly, more work needs to be done to evaluate how well the semantic coherence measure improves the baseline exponential language model. Reduction in perplexity and word error rate should both be determined. Another type of evaluation would be to run a user study to compare our classification accuracy to a person's ability to distinguish between the same sentences.

References

[Cai] Can Cai, Ronald Rosenfeld and Larry Wasserman. Exponential Language Models, Logistic Regression, and Semantic Coherence. In Proc. NIST/DARPA Speech Transcription Workshop, May 2000

[Rosenfeld] Ronald Rosenfeld. A Whole Sentence Maximum Entropy Language Model. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, 1997.

[Zhu] Xiaojin Zhu, Stanley Chen and Ronald Rosenfeld. Linguistic Features for whole Sentence Maximum Entropy Language Models. Proc. Eurospeech '99, Hungary, September 1999.

[Dagan] Ido Dagan, Shaul Marcus, Shaul Markovitch. Contextual Word Similarity and Estimation from Sparse Data. Computer Speech and Language, April 1995

[Cox] Stephen Cox, Shrinandan Dasmahapatra. A Semantically-Based Confidence Measure for Speech Recognition. Proc. ICSLP 2000, China.