Learning Within-Sentence Semantic Coherence

Elena Eneva[†], **Rose Hoberman**[*], and **Lucian Lita**[†]

[†]Center for Automated Learning and Discovery [*]Computer Science Department
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
{eneva,roseh,llita}@cs.cmu.edu

Abstract

In many domains such as speech recognition and machine translation it is extremely useful to be able able to distinguish coherent from non-coherent sentences. We introduce a set of word-based statistical features which measure semantic coherence and can be used to enhance any language application where coherent sentences need to be generated or recognized. We train a decision tree using the constructed feature set to automatically classify sentences as coherent or not. We find that our combination of boosted decision trees and coherence features achieves an accuracy of 80% when distinguishing trigram-generated sentences (non-coherent) from those in the Broadcast News dataset (coherent).

1 Introduction

In order to improve conventional statistical language models, we need to find aspects of language which are not adequately captured, and then incorporate them into the model. Although adding any computable feature of a language which discriminates between the 'true' English language model and the statistical model will give some improvement, an ideal feature should occur frequently, yet exhibit a significant discrepancy (Rosenfeld et al., 2001).

Perhaps the most salient deficiency of conventional language models is their complete failure at modeling semantic coherence. These models capture 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 the 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 *I would like a glass full of water*, and the system recognizes *class* instead of *glass*, we would like to automatically identify and correct this error.

One way to identify and thus avoid these types of errors is to identify sentences which are not semantically coherent. This work is an attempt to automatically construct features which can then be used to learn a function which will distinguish between coherent and non-coherent sentences.

Once we have learned such a measure of semantic coherence, we can combine it with the initial baseline model. One model which is naturally suited for this task is the maximum entropy model introduced by Rosenfeld (1997), which directly models the probability of an entire sentence. The model is ideal for modeling whole-sentence phenomena because it treats each sentence as a 'bag of features' where features are arbitrary computable properties of the sentence.

To incorporate semantic coherence into this exponential model, (Cai et al., 2000) did some initial work on constructing within-sentence semantic coherence features. A set of five features was extracted from each sentence and added directly to the exponential model. These features decreased perplexity over the baseline trigram model by 3 to 5%.

In this paper we augment the limited feature set presented in (Cai et al., 2000) by constructing roughly seventy features, which fall into nine broad classes. The majority of the fea-

[*] Supported by a National Science and Engineering Graduate Fellowship

Table 1: Yule's Statistic

		WORD 1 YES	WORD 1 NO
WORD 2	YES	C_{11}	C_{12}
WORD 2	NO	C_{21}	C_{22}

ture classes rely on word-pair correlation values in sentences. To derive a measure of whole sentence coherence, we used our feature set to build a classifier which distinguishes between real sentences generated by a human (taken as examples of coherent sentences) and fake sentences generated by a conventional language model (taken as examples of non-coherent sentences). The confidence score produced by our classifier has an accuracy of 80%, which is a significant performance improvement over the results obtained by using only the features developed in previous work. This measure of semantic coherence can be used to improve the original language model, and thus improve performance in any application where language modeling is used.

2 Approach

We use semantic association between word-pairs and other sentence characteristics (described later) to construct useful features for distinguishing between coherent and non-coherent sentences.

2.1 Semantic Association between Word-Pairs

We use the same correlation measure as in (Cai et al., 2000). For each word-pair in our dataset, we calculate a measure of association called Q (Yule's statistic) based on the appropriate 2×2 contingency table of training data co-occurrences of the two words in the pair. Table 1 shows such a contingency table for two words. 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 using equation 1.

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

The values of Q thus 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, coherent or non-coherent, can be represented by a variable length list of Q values–a Q value for each pair of content words. 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 Features

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 use syntactic knowledge, information about specific words, or lexical resources like thesauri. The features are based only on the word-pair correlation values, the co-occurrence counts, and statistics on stop words. We have implemented roughly seventy features, which fall into nine classes. These classes are:

1. Simple statistics - the simple correlation characteristics of a sentence: mean, median, maximum, minimum, range and variance of word pair correlation values. These fall into two sub-groups:
   (a) Statistics considering all word-pairs
   (b) Statistics considering only pairs separated by at least 5 words - short distance correlations and local coherence are usually well modelled by trigram language models
2. Sentence Length
3. Percentage of correlation values above some threshold.
4. High/low correlations across large/small distances - very high correlations across large distances are strong evidence that the sentence is from Broadcast News, as are negative correlations at very close distances. We use thresholds to vary what is

considered a high correlation or large distance.

5. Number of word clusters - the number of topics in a sentence can be approximated by clustering the content words in the sentence. We use a greedy, agglomerative clustering algorithm, where similarity is defined as the largest word-pair correlation between any pair of words from the two different clusters. We stop merging clusters when the maximum similarity is smaller than a threshold, which ranged from 0 to 0.8.
6. Word and phrase repetition - in real sentences content words are often repeated, as are some phrases. We calculate the number of repetitions, as well as the length of the longest repeated phrase.
7. Unseen pairs - trigram-generated sentences are much more likely to include pairs of words that were never seen together in a large training corpus. We use both the number and percent of unseen pairs.
8. Content and Stop Words - the percentage of words in the sentence which are content words, the percentage of stop words, and the longest consecutive sequences of content words and of stop words.
9. Likelihood Ratio - the likelihood that a sentence came from a coherent source divided by the likelihood the sentence was trigram-generated. This feature is described in more detail in the following section.

2.3 Likelihood Ratio

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 will generally be semantically similar, but as the words get farther apart the effects of the trigram will weaken, and the association diminish. In real sentences, on the other hand, we hypothesize that the semantic correlation between word-pairs should decrease more slowly with distance.

We tried to model this behavior by generating density estimates (for each corpus) of the Q values conditioned on the distance between the words. The distance is defined so that adjacent words are considered to be at a distance of one. Kernel density estimation was used to smooth the density function.

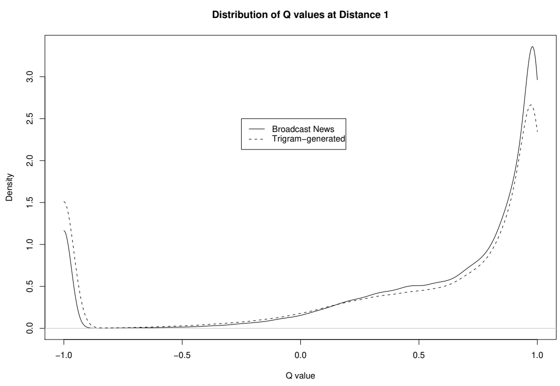

Figure 1: Distribution of Q values for word-pairs at a distance of 1

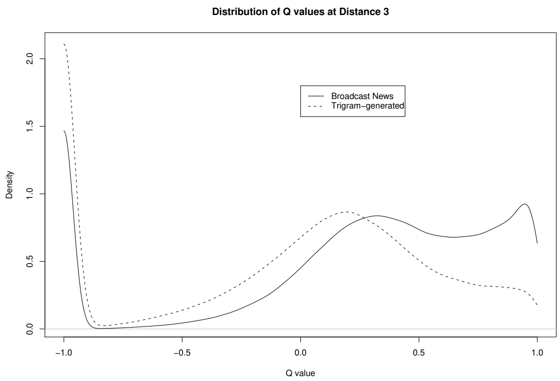

Figure 2: Distribution of Q values for word-pairs at a distance of 3

Figures 3 and 2 show the distributions of the Q values in Broadcast News and in trigram-generated sentences for word- pairs at distances of one and three respectively. As expected, these distributions are almost identical for adjacent words, but outside the trigram model (distance 3), there are significant differences.

We used the generated density estimates to derive a feature, which we call the likelihood ratio. Each sentence is represented as a set of correlation/distance pairs: $(Q_1, d_1), (Q_2, d_2), ..., (Q_n, d_n)$ where n is the number of content word-pairs in the sentence, Q_i is the correlation between the i^{th} pair of content words, and d is the distance between the two content words.

From these density estimates we can estimate the probability in each corpus of seeing a word-pair with a correlation of Q at a distance d: $D(Q|d, BroadcastNews)$ and $D(Q|d, Trigram)$. If we assume (incorrectly but for simplicity) that each pair was generated independently we can reduce the likelihood of the sentence to the product of the likelihood of each content word-pair correlation:

$$L(s|BNews) = \prod_{i}^{n} D(Q_i|d_i, BNews)$$

$$L(s|Trigram) = \prod_{i}^{n} D(Q_i|d_i, Trigram)$$

We can then use the ratio of these likelihoods $\frac{L(s|BroadcastNews)}{L(s|Trigram)}$ as a measure that the sentence is a real Broadcast News sentence. For practical purposes we actually calculate $\log L(s|BroadcastNews) - \log L(s|Trigram)$.

If the value of this feature is a very large positive number, we are very confident that the sentence came from Broadcast News, and is thus likely to be semantically coherent. If the ratio is a very large negative number, it is not likely that the sentence was an actual broadcast news sentence, and is therefore most likely not semantically coherent. The closer the feature is to zero, the more uncertain we are in our classification.

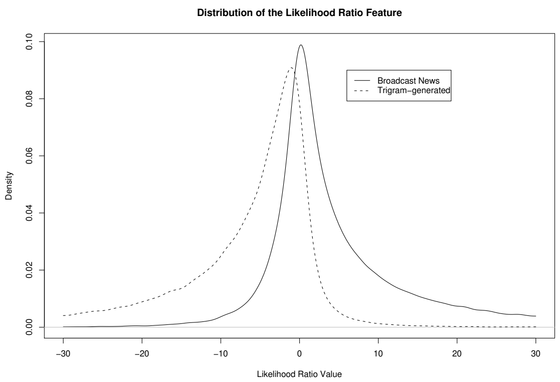

Figure 3: Distribution of the Likelihood Ratio

2.4 Learning Algorithms

We used the features described in the previous sections to train decision trees to classify a sentence as coherent or not.

The decision tree algorithm was C5.0 (an extension of C4.5 proposed by (Quinlan, 1993)), and the selection criterion applied was gain ratio. We set a minimum of twenty training examples per rule, but also tried other values. We also experimented with boosting decision trees, using an implementation of Adaboost.MH (Schapire, 1999).

3 Datasets

Our dataset consists of approximately 104 million words of Broadcast News text (BN). We use 103 million words to estimate the word-pair correlations. We also construct a conventional trigram model from this data using a variation of Kneser-Ney smoothing (Kneser and Ney, 1995). By sampling from this initial trigram distribution we generate an artificial corpus of non-coherent sentences. We combined the remaining 1 million words of broadcast news with an equal number of trigram-generated sentences to get a corpus of approximately 60,000 sentences.

Since we are only concerned with the semantic content of each sentence, we ran experiments with two different stop lists, a small list of the 50 most frequent words and a much larger list of 630 function words. These stop lists removed around 40% and 60% of the tokens respectively. Any word which was not removed we considered a content word.

In general the sentences generated by the trigram model are not coherent. However, since the trigram model does effectively model local behavior, it is incorrect to assume that short sentences produced by the model will not be 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. For this reason, we excluded from the corpus all sentences with fewer than seven words. Also, because most of features look at the correlation between pairs of content words, we discarded any sentence with fewer than two content words.

4 Results and Discussion

For each experiment Table 2 reports the average accuracy and standard deviation after performing 5-fold cross validation on 60,000 sentences. The boosting experiments were all run for 300 rounds, since the performance tended to level

Table 2: Classification Accuracy

Feature Set	Stop Words	DT Acc.	DT StdDev.	Boost Acc	Boost StdDev.
Previous Features	630	72.85	0.92	71.74	0.64
Previous Features	50	74.30	0.58	73.39	0.36
Likelihood Ratio	630	76.20	0.30	76.53	0.65
Likelihood Ratio	50	77.64	0.53	77.76	.49
All but Likelihood	630	76.22	0.68	78.90	0.42
All but Likelihood	50	76.91	0.30	80.37	0.46
All Features	630	77.53	0.43	78.90	0.42
All Features	50	78.73	0.49	80.37	0.46

off after that point.

As can be seen in the table, the best accuracy achieved (80.37%) was when pre-processing with a small stoplist, and using all of our features with Adaboost. This result is a significant improvement over the results obtained by using features developed in previous work (Cai et al., 2000). 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 74.30%.

In general, removing the 50 most frequent words gave better accuracy than using a standard stoplist containing 630 words, and boosting also consistently made slight improvements. Although boosting the decision trees resulted in slightly better performance, this improvement was achieved at the cost of substantially increased runtime.

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

Likelihood_Ratio > 0.93 &
Maximum_Q_at_Distance_4 > 0.915 &
Number_Clusters ≤ 14 ⟹ Coherent_Sentence

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

Figures 4 and 5 show the distribution of two discriminating features. There are significant differences between the distribution of these features in sentences generated by the trigram model and those found in human-generated sentences.

Looking at the rules generated by C5.0 we noticed that the likelihood ratio had the highest information gain among all the features. Using this feature alone gives an accuracy of 77.76%

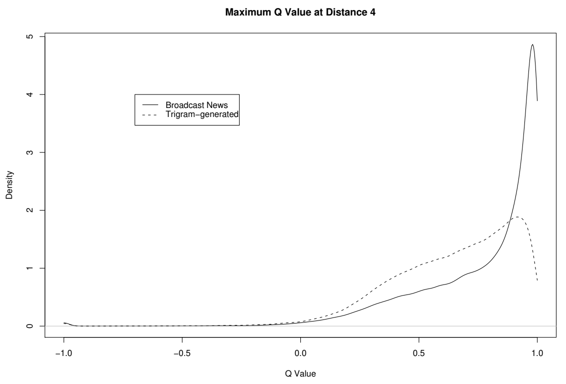

Figure 4: Maximum Q Value at Distance 4

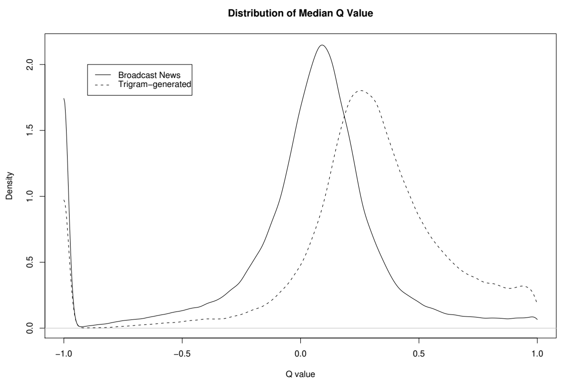

Figure 5: Median Q Value

However, adding the rest of the features only increases accuracy to 80.37%. Most likely this is because there is significant overlap in the information captured by the two sets of features. However, when determining the feature weights an exponential language model takes into account the dependencies among features, and therefore the high degree of overlap should not be of concern.

4.1 Shannon-Style Experiment

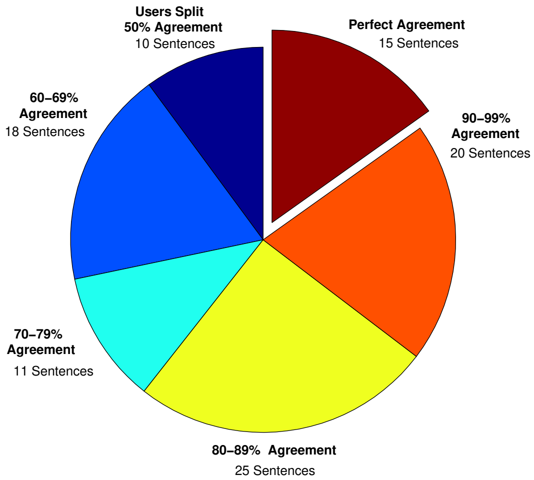

Figure 6: Agreement Among Users

We carried out a Shannon-style experiment to try to determine an upper bound for this discrimination task. We constructed two test sets. Each set was comprised of 50 sentences, evenly divided between Broadcast News and trigram-generated. We use the same pre-processing steps as described in Section 3. Using the 630-word stop list, we replaced all function words with a dash so that only the content words were identifiable.

Each participant was presented with one of these sets and asked to decide whether each sentence was Broadcast News or trigram-generated. Given the reduced version of the sentences, the average accuracy of our 30 participants was 73.77%, with a standard deviation of 6, a median of 74% and a maximum of 84%.

We tested our classifier on the 100 test sentences processed with the same stop list. With the models learned from our earlier experiments, the decision tree trained on all the features achieved 76.16% accuracy with a standard deviation of 1.36. With boosting we reached 78.9% accuracy with a standard deviation of 0.42.

The purpose of this experiment was to compare the success of our classifier to human performance. On these experiments we did not observe statistically significant differences between the two means (within one standard deviation). However, we can say that our classifier performs almost as well as the most accurate of the participants.

On average, 81% of the participants gave the same classification to a sentence. Figure 6 divides the sentences in the experiment into six groups of varying user agreement. The group of sentences with perfect user agreement, for example, consists of those sentences which all the users classified with the same label, regardless of whether their classifications were correct. This group includes the easiest sentences, which everyone got right, as well as the most difficult sentences, which everyone missed.

It is also interesting to compare people's accuracy on specific sentences with the learner's performance on the same sentences. On average, a sentence was classified correctly by the participants of the study 72.65% of the time. For the sentences misclassified by our learner this number was only 60.07%, while for those correctly classified, their accuracy was 76.92%. This suggests that the sentences which the learner misclassifies are also harder for people to label correctly.

4.2 Perplexity Reduction

In addition to reporting classification accuracy, we can also evaluate the expected reduction in perplexity when combining this feature with the baseline trigram model in an exponential language model. The upper bound on improvement from a single binary feature f_i is the Kullback-Leibler divergence between the true distribution ($p(f_i)$) and the distribution according to the baseline model ($p_o(f_i)$) (Pietra et al., 1997). If we estimate $p(f_i)$ with the empirical distribution $\hat{p}(f_i)$, then the probability of the semantic coherence feature being 1 is estimated at .78 for a Broadcast News sentence, compared to .18 for a trigram-generated sentence. We can then compute $\sum_i D(\hat{p}(f_i)||p_o(f_i)) = 1.24$ which translates into an expected perplexity reduction of 4.19% ($2^{\frac{.819}{21}}$, where 21 is the average number of words in a sentence).

Although 4.19% does not appear to be a very large improvement, it is important to remember that this number reflects the average perplexity reduction *per word*. The improvement obtained by adding this feature to a model might be significantly greater on a a task like n-best list rescoring, which requires a whole-sentence judgment. Measuring the magnitude of this improvement remains to be done as future work.

5 Future Work and Summary

Yule's Q statistic is not necessarily the best measure of word-pair correlation because it is extremely susceptible to data sparseness. The point estimates of the Q values for words that appear infrequently in our corpus are often unreliable. One idea for reducing the effect of these unreliable estimates is to use a confidence distribution instead of the current point estimate. Another possible approach would be to use a different correlation statistic; one possibility is the χ statistic. Another possibility is to use the method of (Dagan et al., 1995) to improve the estimates of the likelihood of co-occurrences that are rare in the training data.

When people are given this classification task of distinguishing coherent and non-coherent sentences, they tend to group together words in the sentence that belong to the same topic. We would like to model this process by using other clustering methods to group the semantically similar words together in each sentence. Then statistics based on these clusters could be used as additional features to improve the coherence measure.

In this paper we developed a successful approach for automatically distinguishing between coherent and non-coherent sentences. Our combination of feature sets and learners achieved a reduction in error of 24% from the features used in previous work (Cai et al., 2000). Our successful approach used the semantic association between word-pairs as well as simple features such as sentence length and percentage of stop words to construct a highly accurate classifier.

More work remains to be done to evaluate how well the semantic coherence measure improves the baseline language model. This evaluation can be performed by incorporating the semantic coherence features into an exponential language model and then using the model for a practical application such as n-best list rescoring.

References

C. Cai, R. Rosenfeld, and L. Wasserman. 2000. Exponential language models, logisitc regression, and semantic coherence. In *Proceedings of NIST/DARPA Speech Transciption Workshop.*

I. Dagan, S. Marcus, and S. Markovitch. 1995. Contextual word similarity and estimation from sparse data. *Computer Speech and language.*

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Converence on Acoustics, Speech, and Signal Processing.*

S. Della Pietra, V. Della Pietra, and J.Lafferty. 1997. Inducing features of random fields. In *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

J. R. Quinlan. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA.

Ronald Rosenfeld, Stanley F. Chen, and Xiaojin Zhu. 2001. Whole-sentence exponential language models: a vehicle for linguistic-statistical integration.

R. Rosenfeld. 1997. A whole sentence maximum entropy language model. In *Proceedings of IEEE Workshop on Automated Speech Recognition and Understanding.*

Robert E. Schapire. 1999. A brief introduction to boosting. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.*