# Topic Classification for Conversational Speech using Support Vector Machines and Latent Semantic Analysis

**Dustin Hillard**
University of Washington, EE
hillard@ee.washington.edu

## Abstract

Support vector machines and latent semantic analysis are implemented to develop a topic detection system for conversational speech. Conversations are represented by a vector with word features similar to mutual information. Latent semantic analysis is utilized to transform to a smaller feature space for classification with linear support vector machines. Pair wise voting of topic versus topic support vector machines is used to determine a final topic classification.

## 1 Introduction

This work investigates topic classification for conversation telephone speech. Given a closed set of topics and training conversations, test conversations with an unknown topic are classified as a single topic. A typical task for past research has been topic spotting for newswire data. This work departs from most standard topic classification research because it deals with conversational speech, which has different language patterns and characteristics.

Numerous techniques for topic classification have been well documented. In this work support vector machines (SVMs) are chosen due to their relatively strong performance on a wide variety of domains and tasks. Support vector machines are a natural fit for topic classification because they deal well with sparse data and large dimensionality.

Latent semantic analysis (LSA) is an additional enhancement that can further improve the feature representation of conversations. LSA uses singular value decomposition to decompose the sparse conversation by term matrix and provides a method for extracting a feature space which can represent a higher order structure with linear combinations of word features, rather than just the individual words.

The methods of this research build on previous related work in topic classification as described in Section 2. The approach to classifier design is developed in Section 3. Experimental results are detailed in Section 4, and the main conclusions of this work are summarized in Section 5.

## 2 Related Work

Topic classification is a well researched task in which many different approaches have been shown to be effective. Different methods such as regression models, nearest neighbor classification, Bayesian probabilistic approaches, decision trees, inductive rule learning, neural networks, on-line learning and support vector machines are reviewed by (Yang and Liu, 1999) and (Sebastiani, 2002). Among these approaches SVMs consistently perform well for various tasks and domains. Due to the broad success of SVMs they are chosen for this work.

### 2.1 Support Vector Machines

Support vector machines were introduced in (Vapnic, 1995) and basically attempt to find the best possible surface to separate positive and negative training samples. The 'best possible' means that surface which produces the greatest possible margin among the boundary points. SVMs were developed for topic classification in (Joachims, 1998). Joachims motivates the use of SVMs using the characteristics of the topic classification problem: a high dimensional input space, few irrelevant features, sparse document representation, and that most text categorization problems are linearly separable. All of these factors are conducive to using SVMs because SVMs can train well under these conditions. That work performs feature selection with an information gain criterion and weights word features with a type of inverse document frequency. Various polynomial and RBF kernels are investigated, but most perform at a comparable level to (and sometimes worse than) the simple linear kernel. A software package for training and evaluating SVMs is available and described by (Joachims, 1999). That package is used for these experiments.

### 2.2 Latent Semantic Analysis

Another successful technique for topic classification problems has been latent semantic analysis. The idea was first explored for document indexing in (Deerwester et al., 1990). That work was interested in addressing the issues of synonymy (there are many ways to refer to the same idea) and polysemy (most words have more than one distinct meaning). These two issues present problems for topic classification because two conversations about the same topic need not contain the same words, and conversely two conversations about different topics may contain the same words (but with different intended meanings). In order to find a different feature space which avoids these problems, a singular value decomposition (SVD) is performed to derive orthogonal vector representations of documents. SVD uses eigen-analysis to derive linearly independent directions of the original term by document matrix. SVD decomposes the original term by document matrix $X$ which has dimensions $t$ x $d$, into three other matrices:

$$X = T_0 S_0 D_0^t \tag{1}$$

In equation 1 the matrices $T_0$ ($t$ x $m$) and $D_0$ ($d$ x $m$) have orthonormal columns, and $S_0$ ($m$ x $m$) is the diagonal matrix of singular values as shown in Figure 1.

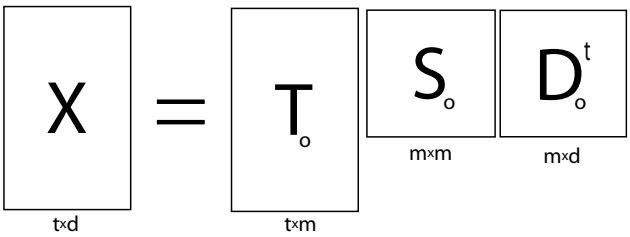


Figure 1: SVD of term by document matrix X

The representation is essentially unique if the singular values are constrained to be positive and decreasing along the diagonal. The inner dimension $m$ is the smaller of $t$ and $d$ and the reduction in dimensionality comes from selecting only the top $k<m$ singular values and associated eigenvectors. This yields a new equation which provides the lowest least squares error approximation to $X$:

$$X \approx X' = TSD^t \tag{2}$$

In equation 2 the matrices $T$ ($t$ x $k$) and $D$ ($d$ x $k$) are reduced from their original form, and $S$ ($k$ x $k$) is the diagonal matrix of $k$ singular values. This approximates the original term by document matrix with the $k$ largest singular value triplets as shown in Figure 2.

Various comparisons among terms and documents can be investigated with dot products between the eigenvectors from $D$ and $T$. For this work the primary desire is a document to document comparison for topic classification. The matrix $X$ can be considered as the original training data, and documents are represented by

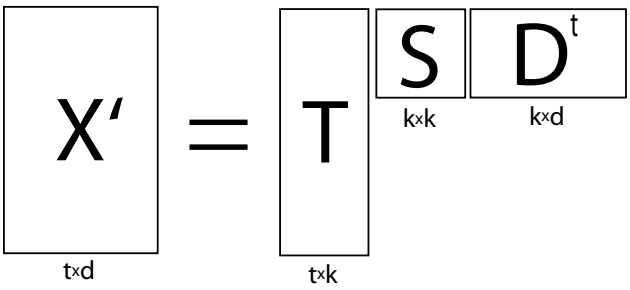


Figure 2: MLSE representation of X with top $k$ singular value triplets

the rows of $DS$. An equivalent transformation for new test vectors is also desired, and solving the equation for the terms $DS$ with a new test matrix $Y$ of terms by documents gives:

$$DS = Y^t T \tag{3}$$

The rows of this matrix can then be compared with those from the training set because the word features have undergone the same transformation from matrix $T$. After obtaining the LSA feature vectors a classifier can then be trained on those features. The initial work of (Deerwester et al., 1990) proceeded to just find a minimum cosine distance between vectors, while (Cristianini et al., 2001) elegantly implemented LSA as a kernel for SVMs. Implementing the LSA kernel for SVMs was beyond the scope of this project, but the LSA feature vector can be used as input features to train SVMs.

### 2.3 Word feature processing

Finally, text input to topic classification systems is usually preprocessed and then word features given weights depending on importance measures. Most text classification work starts with word stemming to remove variable word endings and reduce words to a canonical form so that different word forms are all mapped to the same token (which is assumed to have essentially equal meaning for all forms).

Word features usually then consist of stemmed word counts, adjusted by some weighting. Inverse document frequency is commonly used, and has some justification (Papineni, 2001). More complex measures of word importance have shown to provide additional gains though. A weighted inverse document frequency is an extension of inverse document frequency to incorporate term frequency over texts, rather than just term presence (Tokunaga and Iwayama, 1994).

Term selection can also help improve results and many past approaches have found information gain to be a good criteria (Yang and Liu, 1999) and (Sebastiani, 2002).

## 3 Classifier Design

Classifier design for this project combines proven methods and additionally implements some new ideas to seek further improvements. The classifier will be described in the sequence that training and testing data are processed in experiments.

### 3.1 Word feature processing

The first step in preparing the text for feature extraction is to remove all non-word tokens and map everything to lower case, so that the remaining text consists of only lower case words with no punctuation or extraneous tokens that may exists in the raw text. The following step performs the standard Porter Stemming Algorithm (Porter, 1980). Statistics of term occurrence are then computed, finding the number of occurrences of each term per conversation. Frequencies for bigrams (word pairs) and trigrams (triples) are also computed for use as features spanning multiple adjacent words. While overall statistics are computed for all words, bigrams and trigrams are collected after removing the 150 words with the lowest mutual information for the corpus to reduce the occurrence of bigrams that would be redundant to with single words (e.g. *a dog* and *dog*).

The counts are normalized by conversation length because lengths differ from less than 1000 to more than 2000 words, which seems to indicate that longer conversations may have more occurrences of content words (while it

may also be true that a speaker may talk a lot without using content words, such as many instances of *yeah*, *like* or *you know*).

Topic frequencies are then computed by averaging term frequencies from the individual conversations, and and overall corpus frequency is computed by averaging the topic frequencies. Although typical word feature weightings such as inverse document frequency have been generally effective, as mentioned previously, more detailed forms of word weighting have also provided further gains. This work adopts a weighting related to mutual information, where each word is given a feature value $w_i$ as shown in Equation 4.

$$w_i = log(\frac{p(w,t)}{p(w)p(t)}) = log(\frac{p(w|t)p(t)}{p(w)p(t)}) \quad (4)$$

This also reduces to an intuitive form as in Equation 5 where it can be thought of as a ratio of frequency given a topic divided by the overall frequency in the corpus.

$$w_i = log(\frac{p(w|t)}{p(w)}) \quad (5)$$

Finally, only words with $w_i$>0 are placed in the term by conversation matrix (this is all terms with a ratio greater than 1, or in other words those that occur more frequently than the corpus average).

### 3.2 Latent Semantic Analysis Implementation

The processing of Section 3.1 produces a term by conversation matrix that is the $X$ (from all training conversations) to be decomposed as described in Section 2.2. The size for dimensionality reduction is chosen empirically based on a development set and is discussed in Section 4. Document feature vectors for the training set are then selected as the rows of matrix $DS$ and document feature vectors for the test set are selected as rows of matrix $Y^tT$ as described in Section 2.2. These documents feature vectors are then supplied to SVM classifiers.

### 3.3 Support Vector Machine Implementation

Support vector machines are trained with feature vectors from the LSA output. An SVM is trained for each pairing of topics so that a classifier to distinguish between each binary pairing is available. Linear SVMs are utilized because past work has shown that text categorization is almost always linearly separable, and more complex kernels such as polynomials or RBFs have shown little gains for this task. In addition, more complex kernels introduce additional variability due to the required tuning of additional kernel parameters.

The testing condition evaluates a test feature vector against each of the binary classifiers, with each classifier voting for the topic that is predicted by that individual SVM. The final topic is selected by choosing the topic that has the greatest number of votes from the committee of classifiers. Additional methods of classifier combination were explored, but no gains were realized within the time frame of this project. The software package $SVM_{light}$ is used to train and evaluate all support vector machines.

## 4 Experiments

### 4.1 Experimental Setup

A training set of over 750 conversations was used in conjunction with a development set of almost 600 conversations for system development and tuning experiments. This training set size was sufficient for preliminary testing, and in addition a large development set was useful to allow more confidence in development experiments. The final system is trained using all training and development data available in order to maximize the amount of data used for LSA and SVM training. The final test set consisted of about 900 conversations and overall results are reported for testing on this evaluation set.

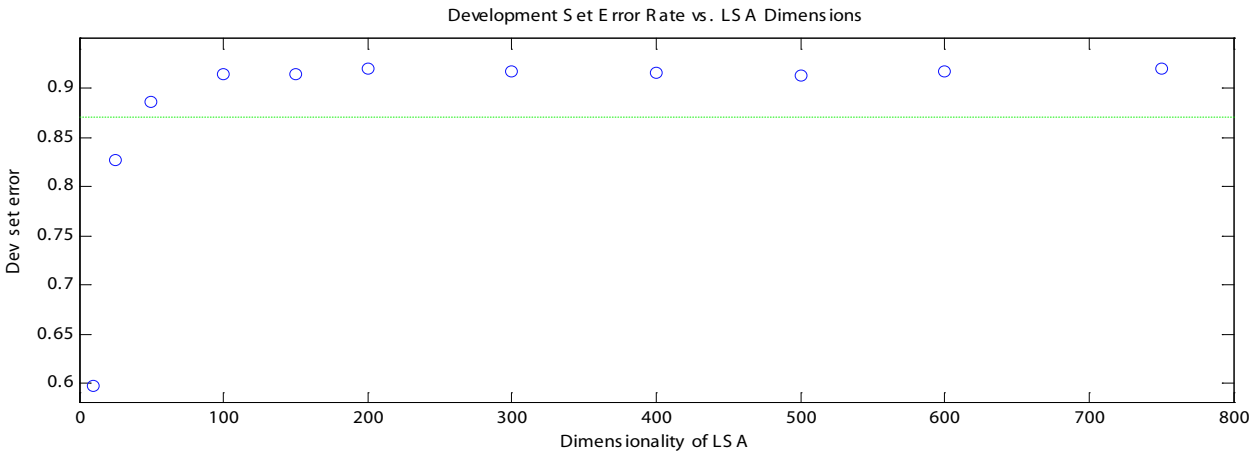


Figure 3: Classification error rate versus LSA dimensionality

### 4.2 Results

Initial experiments tested the parameter used for feature selection for the term by conversation matrix. Experiments confirmed that the intuitive cutoff of zero for the word features $w_i$ was optimal. Error rates increased for both higher and lower cutoffs, so the cutoff was selected for all future experiments due to experimental success and intuitive benefit.

The primary domain for parameter tuning was dimensionality of the SVD for latent semantic analysis. Training on the training set and testing on the development set, the error rates of Figure 3 were obtained. The dotted green line represents performance without LSA, meaning that SVMs are trained on the raw $w_i$'s. The results for LSA are somewhat surprising given expectations based on previous work. Most LSA work describes systems using about 100 dimensions, but for these experiments results continued to improve with increases in the LSA dimensionality. The typical range of about 100 dimensions does give similar performance to the higher dimensionalities, but there is a small gain for the larger LSA (and this trend continues with further results on the test set presented at the end of this section). The figure shows results ranging from 10 to 750 dimensions, further increases were not possible because the SVD is limited to the size of the smallest dimension (in this case the approximately 750 training conversations).

The best performing system in development had the highest dimensionality, so the final evaluation system, when trained on all training and development data used an LSA dimensionality of 1250 (possible because the number of conversations was larger and the size of the SVD could be increased).

Results for the evaluation set are given in Table 1 below. The left column describes the conversations used for training, where 'train' is just the 750 training conversations, and 'train+dev' is the 1250 training and development conversations. The difference between row one and two manifests the performance gain from increasing the training set size, which is .7%. The change from row two to three shows the performance gain from increasing the dimensionality of the LSA from 750 to 1250, which results in 1.1% absolute improvement.

| Train Set | LSA Dims | Accuracy |
|---|---|---|
| train | 750 | 92.8% |
| train+dev | 750 | 93.5% |
| train+dev | 1250 | 94.6% |

Table 1: Results for the evaluation set

A final set of experiments (that was not completed in time for the official evaluation) expanded the vocabulary to include bigrams and trigrams as explained in Section 3.1. These further experiments investigated the value of word sequences in topic classification, the results are provided in Table 2.

| Train Set | LSA Dims | Words | +n-grams |
|---|---|---|---|
| train+dev | 750 | 93.5% | 94.8% |
| train+dev | 1250 | 94.6% | 94.5%* |

Table 2: Results for the evaluation set (including bigrams and trigrams)

Adding bigrams and trigrams provides the overall best accuracy across all systems. For LSA with 750 dimensions the accuracy improved by 1.3% absolute when adding bigrams and trigrams to the vocabulary. The result for 1250 dimensions is not directly comparable because the SVD package used for all other experiments failed for the large number of conversations and increased vocabulary size. The result for 1250 dimensions with n-grams did improve (about 1%) when compared to the case of 750 dimensions with n-grams when using the same SVD computation algorithm.

## 5 Conclusion

A topic classification system for conversational telephone speech is developed that obtains 95% accuracy for a set of 67 topics. Word features are a mutual information measure, and all words in a particular conversation occurring less often than the corpus average are left out. Bigrams and trigrams are added to the vocabulary (after removing low information words) and provide gains when compared to the simple vocabulary of single words. Latent semantic analysis improves results when compared to simple word features, and performs best at a dimensionality of 750 to 1250. A committee of binary topic classification support vector machines trained on LSA feature vectors selects a final topic by majority voting. Increasing training data size also led to better accuracy. Future improvements could be investigated with additional experiments varying vocabulary size, word feature indexing, and LSA/SVD computation.

[1]Result is not directly comparable because SVD was implemented with Matlab function svd() while all other results used svds(), which provided somewhat better results. The 750 dimension +n-gram case had accuracy 93.7% when using svd() rather than 94.8% with svds(). Numerical precision differences between the functions seem to be to blame, although it is surprising that such a strong effect occurs (i.e. $>$1%).

## References


N. Cristianini, J. Shawe-Taylor, and H. Lodhi. 2001. Latent semantic kernels. In C. Brodley and A. Danyluk, editors, *Proceedings of ICML-01, 18th International Conference on Machine Learning*, pages 66–73. Morgan Kaufmann Publishers, San Francisco, US.

S. Deerwester et al. 1990. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*.

T. Joachims. 1999. Advances in kernel methods - support vector learning. *Making large-Scale SVM Learning Practical*.

K. Papineni. 2001. Why inverse document frequency? In *NAACL Proceedings*, pages 25–32.

M. F. Porter. 1980. An algorithm for suffix stripping. *Program*, 16(3):130–137.

F. Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1).

T. Tokunaga and M. Iwayama, 1994. *Text categorization based on weighted inverse document frequency*. Technical Report 94 TR0001, Department of Computer Science, Tokyo Institute of Technology.

V. Vapnic. 1995. *The Nature of Statistical Learning Theory*. Springer, New York, NY.

Y. Yang and X. Liu. 1999. A re-examination of text categorization methods. In *Proceedings of SIGIR-99*, November.