# MINING SPEECH: AUTOMATIC SELECTION OF HETEROGENEOUS FEATURES USING BOOSTING

*Aldebaro Klautau*

ECE Department, UCSD
9500 Gilman Drive
La Jolla, CA 92093, USA

**ABSTRACT**

We investigate feature selection applied to automatic speech recognition (ASR) systems. We focus on systems based on support vector machines (SVM), which can naturally use features optimized for each classifier. We present a new method for feature selection based on the AdaBoost algorithm. This method was an order of magnitude faster than a similar one, while leading to equivalent accuracy. Experiments with phone classification using TIMIT and a total of 760 features (PLP, MFCC, Seneff's, formants, etc.) indicated that the proposed method automatically discovered important information in the data. When using only 25 selected features per SVM, the accuracy was higher than when using a homogeneous set of 118 features based on PLP coefficients.

## 1. INTRODUCTION

Front ends in ASR are typically based on mel-frequency cepstrum coefficients (MFCC) [1] or perceptual linear prediction (PLP) coefficients [2]. Both were designed based on existing knowledge about speech communication. For example, psychoacoustic experiments with pitch perception and tone masking led to the mel and Bark frequency scales, which are used for calculating MFCC and PLP coefficients, respectively. We call such features *knowledge-based*.

Specifying knowledge-based features is time-consuming at best, and impossible in many domains. For this reason, it is important to investigate whether *data-driven* approaches to feature extraction can be competitive with knowledge-based ones. In addition, even when experts are available to propose a set of knowledge-based features, there is usually no guarantee that this set leads to optimal or near-optimal performance. When this is the case, it is important to try data-driven features, to evaluate whether the knowledge-based features are adequate. This work investigates data-driven methods applied to the design of front-ends in ASR. We use automatic feature selection methods [3] to mine relevant information in the speech signal.

Another aspect in which the front ends investigated in this work departs from typical ones is that we allow for heterogeneous features. Heterogeneous features are an interesting topic because even a well-tuned MFCC or PLP front end cannot optimally represent all phonetic classes. For example, nasal and stop sounds have conflicting requirements, with a nasal requiring improved frequency resolution and a stop requiring time resolution [4].

Heterogeneous features were previously used in, e.g., [4]. Integrating heterogeneous features and generative models (e.g., HMMs) often requires extra work. For example, the Baum-Welch algorithm had to be modified to support heterogeneous features in [5]. The extra complexity may be the main reason for the relatively small popularity of heterogeneous features when compared, for example, with multi-stream front ends [6]. On the other hand, acoustic modeling based on discriminative classifiers (e.g., SVMs [7]), adds a new degree of freedom when designing ASR systems, because these classifiers naturally support heterogeneous features. An efficient way of obtaining a multiclass classifier from SVMs is through the all-pairs matrix [8]. In this case, it is intuitive that the best features for distinguishing a pair composed, for example, by a vowel and a stop, should differ from the ones to distinguish a pair of vowels. We explore this issue in our experiments.

This paper is organized as follows. In Section 2 we discuss heterogeneous features in ASR an establish the notation. Section 3 introduces the proposed feature selection method based on boosting, and also describes a baseline method based on the information-gain. The experimental results are presented in Section 4, followed by our conclusions.

## 2. HETEROGENEOUS FEATURES

We assume a frame-based front end to simplify the discussion in this section. A slightly more complex notation would be required to take in account, e.g., front ends for segmental modeling.

A typical *frame-based* front end converts *segments* $\mathbf{s}$ of raw speech data into feature vectors $\mathbf{x} = (x_1, \ldots, x_L)$, where $x_i \in \mathcal{X}_i$ and $\mathbf{x} \in \mathcal{X}^L = \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_L$. The conversion occurs at a rate $r$, and $1/r$ is the *frame* duration. Typically, $r = 100$ Hz and $L = 39$ features.

We say the feature set $\mathcal{X}^L$ is *homogeneous* with respect to a given SVM-based ASR system, if the $L$ features compose the training sets of all SVMs. The set $\mathcal{X}^L$ is called *heterogeneous* with respect to the ASR system if there are at least two distinct subsets of $\mathcal{X}^L$ that are used for training SVMs. Similarly, when the acoustic model is based on HMMs, $\mathcal{X}^L$ is heterogeneous with respect to the system if distinct subsets of $\mathcal{X}^L$ are used as input space of HMM output distributions (e.g., when the features depend on the phonetic class of the HMM model).

Note the distinction between *multi-stream* [6] and heterogeneous features. For example, in [9], the set of features was obtained through different signal processing algorithms, but we call it homogeneous because the $L$ features were always used by the acoustic model.

A *heterogeneous front end* consists of a mapping $F : \mathbf{s} \rightarrow \mathbf{x}$ from speech segments to features, and a list of sets with the

Sponsored by CAPES, Brazil.

selected indices $\{\mathcal{I}_b\}$, which determine the subsets of $\mathcal{X}^L$ to be used. In SVM-based ASR, the $b$-th SVM is trained using $L_b = |\mathcal{I}_b|$ features.

A pertinent question is why not always using the $L$ features? For example, the SVMs themselves could identify the unimportant features and deemphasize their influence. In practice, besides decreasing computational cost, reducing the number of features can lead to better generalization capability when data is scarce [10]. Hence, our goal is to reduce $L_b$ while retaining enough information to achieve high accuracy with the correspondent $b$-th SVM classifier.

The methods to reduce the dimensionality of $\mathcal{X}^L$ can be organized into two groups: *selection* and *extraction* [3]. Feature selection methods try to identify and keep only those features that contribute most to the task. These methods do not modify the features, but choose a subset from all $2^L$ possible subsets. Feature extraction methods transform the original space $\mathcal{X}^L$ into a lower-dimensional space, modifying the original features. An example is the use of principal component analysis (PCA) [3] followed by truncation of unimportant components.

In this work we design the heterogeneous front ends using a data-driven approach that consists in extracting a large set of (possibly redundant) features, and then reducing it in a second stage through feature selection. This approach was successfully used for image retrieval in [11]. The next section describes the specific methods we used for feature selection.

## 3. FEATURE SELECTION

For convenience, we discuss feature selection assuming an SVM-based ASR system. The training set $\mathcal{T} = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}$ of each SVM contains $N$ *examples*.[1] Each example $(\mathbf{x}, y)$ consists of an instance $\mathbf{x} \in \mathcal{X}^L$ and a label $y \in \{-1, 1\}$.

In this standard classification framework, feature selection methods can be split into two groups: *filters* and *wrappers* [12]. The latter consists of methods that select features using the learning algorithm $\mathcal{L}$ that will be applied to train the classifier (SVM in this case). In most cases, it is unfeasible to let the wrapper explore all $2^L$ possible subsets of $\mathcal{X}^L$ (i.e., train $2^L$ SVMs). Therefore, wrappers often adopt a heuristic (and possibly suboptimal) search [12]. Filter methods evaluate the worth of features $x_i$ by using heuristics as, e.g., the correlation of $x_i$ with label $y$. Therefore, filters usually demand less computation than wrappers and are independent of $\mathcal{L}$ (i.e., a filter applied to SVM-based ASR does not require to train SVMs int the feature selection stage). On the other hand, wrapper methods can achieve higher accuracy [12]. Here we use only filters, but the full version of this paper presents an evaluation of wrappers for feature selection in ASR.

The first filter method we used is the information gain (also known as mutual information), which is one of the simplest but yet popular method for feature selection [12]. It works as follows. A distribution $p(y)$ on the labels is estimated using the training set $\mathcal{T}$, and the associated random variable $Y$ has entropy

$$H(Y) = -\sum_{y \in \mathcal{Y}} p(y) \log_2 p(y).$$

We assume the features $x_i$, $i \in \{1, \ldots, L\}$, were originally discrete or previously discretized (we used the method proposed in [13]).

[1] We simplify the notation by dropping the subscript $b$ that has been used to identify a specific SVM.

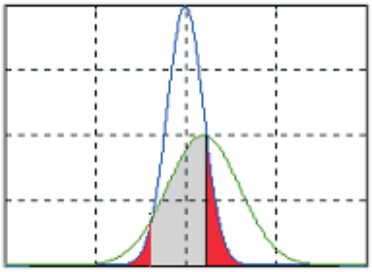

**Fig. 1**. Assuming a uniform prior, The sum of the filled areas corresponds to half of the Bayes error ($\epsilon$ in the proposed method) for these two Gaussians.

After observing $x_i$, the training set can be partitioned according to its value, and distributions $p(x_i)$ and $p(y|x_i)$ estimated by counting occurrences in $\mathcal{T}$. The entropy of $Y$ conditioned on observing the random variable $X_i$ is

$$H(Y|X_i) = -\sum_{x_i \in \mathcal{X}} p(x_i) \sum_{y \in \mathcal{Y}} p(y|x_i) \log_2 p(y|x_i),$$

and the information gain $I(X_i; Y)$ is given by

$$I(X_i; Y) = H(Y) - H(Y|X_i).$$

The method consists in calculating $I(X_i; Y)$, for $i = 1, \ldots, L$, and then selecting the $S$ features with highest $I(X_i; Y)$. The information gain provides a baseline for comparison with the following method.

We propose a new filter method based on the AdaBoost algorithm [14], and inspired by the application of AdaBoost to feature selection in [11]. The algorithm is described in the full version of this paper and here, due to lack of space, we present only an outline. The desired number $S$ of features to be selected is specified in advance and, for each iteration $j = 1, \ldots, S$ of boosting, a feature $x_i$, $i \in \mathcal{J}_j$, is chosen and its index is kept $k_j = i$. For the first iteration, $\mathcal{J}_1 = \{1, \ldots, L\}$ and, for subsequent iterations, the feature that was previously selected is eliminated from further consideration, namely, $\mathcal{J}_{j+1} = \mathcal{J}_j - \{k_j\}$.

AdaBoost keeps a distribution $\mathbf{P}_j$ over $\mathcal{T}$, and tries to focus on hard-to-classify examples $(\mathbf{x}^n, y^n)$ by increasing their weight $\omega_j^n$. This distribution is used to calculate the expected average error $\epsilon_i$ associated to choosing the $i$-th feature. The chosen feature corresponds to the one that minimizes $\epsilon_i$, namely,

$$k_j = \arg\min_{i \in \mathcal{J}_j} \epsilon_i.$$

In the method proposed in [11], $\epsilon_i$ is the average (weighted according to $\mathbf{P}$) training error rate of the best *decision stump* associated to feature $i$ (i.e., the stump that minimizes $\epsilon_i$). This method is time-consuming when both numbers of features and training examples are large. In the proposed method, which is restricted to binary problems, $\epsilon_i$ is calculated as follows.

At each iteration $j$ and for each feature $i$, two Gaussian distributions $\mathcal{N}(\mu_+^i, \sigma_+^i)$ and $\mathcal{N}(\mu_-^i, \sigma_-^i)$ are fit to the examples with *positive* and *negative* labels, respectively, taking in account the current distribution $\mathbf{P}_j$. For instance,

$$\mu_+^i = \frac{1}{|\mathcal{P}|} \sum_{n \in \mathcal{P}} \omega_j^n x_i^n,$$

where $\mathcal{P} = \{n : y^n = +1\}$ is the set of positive examples. The error $\epsilon_i$ corresponds to the *Bayes error* assuming the Gaussians are the true distribution of the two classes, as illustrated in Figure 1. In this case, calculating $\epsilon_i$ requires estimating the Gaussians, solving a quadratic equation to find the points where they cross, and four calls to a routine that calculates the *error function*.

We modified AdaBoost.M1 [15] in Weka [16] to implement the methods proposed here and in [11]. Our method was faster by an order of magnitude, while leading to equivalent accuracy.

## 4. EXPERIMENTAL RESULTS

In this section we evaluate the application of feature selection methods to the design of heterogeneous front ends. We conducted experiments with phone classification using the TIMIT dataset. For training and testing we collapsed the 61 phones into the standard $K = 39$ phonetic classes defined by Kai-Fu Lee [17]. As usually done, the *sa* sentences were excluded from the training set and we report results for the *core* test set.

The set $\mathcal{X}^L$ was obtained from seven streams of features, corresponding to popular front ends in ASR. The combination of these streams leads to a highly redundant set of features. The selection methods should be able to automatically identify the features with complementary information.

The first stream, called *formants*, consisted of F0, probability of voicing and the first four formant frequencies (F1-F4). These features were obtained using pitch and formant estimation algorithms developed by David Talkin and others.[2]. The second stream corresponds to a typical PLP front end [2] with 39 features. This *plp* stream consisted of 13 static coefficients (12 PLP coefficients and energy), and their first and second derivatives. The third and fourth streams consisted of 40 parameters each, obtained from the *synchrony* and *envelope* stages of Seneff's auditory model [18]. Similarly to the *plp* stream, the fifth and sixth streams, consisted of 39 parameters of MFCC [1] and RASTA [19] front ends, respectively. The last stream, with 50 features and called *filter-bank*, was composed by the outputs of 24 filters spaced according to the mel-scale, normalized energy and their first derivatives. The total number of features was 253.

The SVMs are static classifiers and work with fixed-length vectors. In ASR, SVMs are trained using features obtained from frame-based [20] or segmental [21, 22] front-ends. In this work we used the latter approach, with the segmentation being the phonetic transcriptions available in TIMIT.

In HMM-based ASR, phones are often modeled as composed by three stationary parts. Using the same heuristic, the $T$ frames of a phone were split into three sets at a 3-4-3 ratio according to a linear warping [21, 22]. The literature on dynamic-time warping (DTW) (see, e.g., [23]) shows that this is not the ideal warping for speech, but it is attractive due to its simplicity. Averaging the frames of each set led to three subvectors of dimension 253 each.[3] We concatenated the three subvectors and added the *duration* $T$ as the last feature, achieving a feature vector $\mathbf{x}$ with dimension $L = 760$.

The SVMs were organized through an *error-correcting output code* ECOC scheme with the all-pairs matrix and Hamming decoding [8]. Hence, the number of SVMs was $\binom{K}{2} = 741$. In spite of a smaller number of SVMs, the one-versus-rest ECOC matrix leads to a considerable longer training time and accuracy equivalent to all-pairs, as shown in [8]. The binary training sets were normalized, such that the attributes were restricted to the range $[0, 1]$. We were mainly interested on comparing methods, not on achieving the best possible results in terms of accuracy. Therefore, we used linear SVMs due to their shorter training time. We note that, for speech data, the Gaussian (or RBF) kernel often leads to a significantly smaller error than a linear kernel.

| Streams ($L$) | filter ($S$) | Error (%) | distinct features |
|---|---|---|---|
| all 7 + *duration* (760) | proposed (5) | 34.6 | 464 |
| all 7 + *duration* (760) | info. gain (5) | 38.5 | 264 |
| all 7 + *duration* (760) | proposed (25) | 26.6 | 498 |
| all 7 + *duration* (760) | info. gain (25) | 32.1 | 315 |
| all 7 + *duration* (760) | proposed (40) | 25.3 | 755 |
| all 7 + *duration* (760) | info. gain (40) | 30.9 | 532 |
| all 7 + *duration* (760) | proposed (100) | 22.7 | 758 |
| all 7 + *duration* (760) | info. gain (100) | 29.3 | 602 |
| all 7 + *duration* (760) | - | **21.0** | all |
| *plp* + *duration* (118) | - | 28.2 | all |

**Table 1**. Performance of heterogeneous front ends for TIMIT phone classification.

The results are shown in Table 1. In all cases, the proposed method based on boosting outperformed the information gain method in terms of accuracy. The best result, 21% of error, was obtained with all 760 features, i.e., without using feature selection. However, using $S = 100$ features per SVM led to an error of 22.7%. With $S = 5$ features, the error went up to 34.5%. We also counted the number of selected distinct features. When $S = 100$, only two features were never selected. We conducted an experiment using only the *plp* stream and *duration*, which corresponds to $L = 118$. In this case, the error was 28.2% without feature selection, which is worse than the result obtained with $S = 25$ features per classifier.

Even using linear SVMs, our results turned out to be competitive with other systems. For example, in [24], SVMs led to 28.6% of error rate for phone classification using TIMIT.

We now look in more details at the results obtained with $S = 5$ features per classifier. Figure 2 shows the histograms of the selected indices for each stream. For example, for *formants*, only F0 (which is a speaker dependent feature, but indicates voicing and gender) was selected more than 50 times, while F4 was never selected. We obtained the counts for each stream feature, summing the three correspondent features of $\mathbf{x}$. Hence, the total number of bins in all seven histograms is 253. For this system, the streams *formants*, *plp*, *synchrony*, *envelope*, *mfcc*, *rasta*, and *filter-bank* had its most popular feature selected 99, 37, 465, 33, 107, 56, 158 times, respectively. The *duration* feature was the third most popular, being used by 221 SVMs. The average number of times a feature was selected in each stream was 29.0, 6.3, 27.0, 6.2, 16.4, 10.1, and 17.5, respectively. Note that features of *plp* and *envelope* were not selected as often as the others. The *envelope* stream is not very effective for ASR, but that is not the case of *plp*. When we exclude *rasta*, which is relatively similar to *plp*, the system selects more features of *plp* than *mfcc*.

It is interesting to observe that the two features of *synchrony* corresponding to the lowest frequencies, were the only ones to be selected more often than *duration*. Also, the 46-th feature of *filter-bank*, which corresponds to the first derivative of the output of the

[2]These algorithms were part of Waves+, which is not commercialized anymore. They were recently incorporated to the Snack toolkit, which can be downloaded at *www.speech.kth.se/snack*. We used Snack version 2.2.

[3]For some streams the average is taken in the cepstrum domain, which corresponds to a geometric average in the spectrum domain.

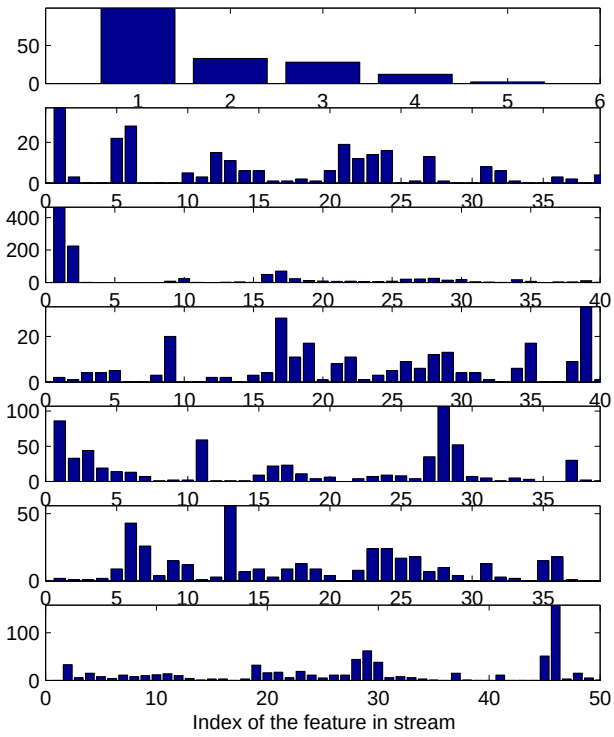


**Fig. 2**. Histograms of selected features for the streams *formants* (top), *plp*, *synchrony*, *envelope*, *mfcc*, *rasta*, and *filter-bank* (bottom).

21-th filter (high center-frequency), was used by 158 classifiers, where the majority was trying to distinguish a fricative sound from others.

## 5. CONCLUSIONS

We presented a new method for feature selection based on the AdaBoost algorithm. The proposed method achieved significantly better accuracy than the popular information-gain method, while being faster. When compared to a set of 118 features based exclusively on PLP coefficients and duration, the heterogeneous front end led to a higher accuracy using 25 features per SVM. The data-driven approach showed interesting relations in the data. Additional experiments are needed to identify the most effective features per phonetic class, and evaluate the heterogeneous front end in other tasks.

## 6. REFERENCES


[1] S. Davis and P. Merlmestein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on ASSP*, pages 357–366, 1980.

[2] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *JASA*, 87(4):1738–52, Apr. 1990.

[3] A. Webb. *Statistical Pattern Recognition*. Oxford University Press Inc., 1999.

[4] A. Halberstadt. *Heterogeneous acoustic measurements and multiple classifiers for speech recognition.* PhD thesis, MIT, 1998.

[5] P. Baggenstoss. A modified Baum-Welch algorithm for hidden Markov models with multiple observation spaces. *IEEE Trans. on Speech and Audio Proc.*, 9(4):411–16, 2001.

[6] H. Bourlard, S. Dupont, and C. Ris. Multi-stream speech recognition. *Jour. Integrated Study of Art. Intell., Cognit. Sci. and App. Epistemology*, 15(3):215–34, 1998.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[8] A. Klautau, N. Jevtić, and A. Orlitsky. On nearest-neighbor ECOC with application to all-pairs multiclass SVM. *Journal of Machine Learning Research*, submitted, 2002.

[9] H. Christensen, B. Lindberg, and O. Andersen. Employing heterogeneous information in a multi-stream framework. In *ICASSP*, volume 3, pages 1571–4, 2000.

[10] A. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *ICML*, 1998.

[11] K. Tieu and P. Viola. Boosting image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 228–235, 2000.

[12] M. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, University of Waikato, 1999.

[13] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *13th International Joint Conf. on Artificial Intelligence*, pages 1022–1027, 1993.

[14] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EUROCOLT*, pages 23–37, 1995.

[15] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *ICML*, pages 148–156, 1996.

[16] http://www.cs.waikato.ac.nz/ml/weka.

[17] K.-F. Lee and H.-W. Hon. Speaker-independent phone recognition using hidden Markov models. *IEEE Transactions on ASSP*, 37(11):1641–8, Nov. 1989.

[18] S. Seneff. Pitch and spectral estimation of speech based on auditory synchrony model. *ICASSP*, 3:36.2/1–4, 1984.

[19] H. Hermansky and N. Morgan. Rasta processing of speech. *IEEE Trans. on Speech and Audio Proc.*, 2(4):578–89, Oct. 1994.

[20] S. Fine, G. Saon, and R. Gopinath. Digits recognition in a noisy environment via a sequential GMM/SVM system. In *ICASSP*, 2002.

[21] P. Clarkson and P. Moreno. On the use of support vector machines for phonetic classification. In *ICASSP*, pages 585–8, 1999.

[22] A. Ganapathiraju. *Support Vector Machines for Speech Recognition*. PhD thesis, Mississippi State University, 2002.

[23] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on ASSP*, 26(1):43–49, 1978.

[24] J. Salomon, K. Simon, and M. Osborne. Framewise phone classification using support vector machines. In *ICSLP*, pages 2645–2648, 2002.