Separating Speech From Noise Challenge

We have used the data from the PASCAL CHiME challenge with the goal of training a Support Vector Machine (SVM) to estimate a noise mask that labels time-frames/frequency-bins of the audio as '*reliable*' or '*unreliable*'. This noise mask could be used by another block in the signal processing pipeline to treat the unreliable data as missing and then replace the missing data with an estimate of the clean audio by searching a corpus of clean audio samples for the most probable match using the features of the unreliable data and the surrounding audio. For this project, we have focused on the noise mask estimation using an SVM and not on the data imputation portion of the problem. It has been demonstrated by Kallasjoki et al. [1] that given a good noise mask, it is possible to achieve significant improvements in automated speech recognition accuracy rates by replacing unreliable portions of the audio with estimates of the clean audio.

In order to judge the SVM classification accuracy in generating a noise mask, we needed an '*oracle mask*' which gave the correct answer for the mask. The oracle mask generates a label of reliable or unreliable for time frame/frequency pairs, using the Mel filterbank energies of the clean signal and the noisy signal and labeling time/frequency pairs as unreliable if the SNR is less than -3 dB.

Using the oracle mask, we estimated the best-case performance of noise mask estimation and data imputation by replacing unreliable time/frequency segments (as labeled by the oracle mask) with the known clean speech audio. As expected, this achieves very good recognition rates that approach the accuracy of the using the clean speech audio. As previously discussed, previous work [1] has demonstrated significant performance improvements using an oracle mask along with sparse imputation methods. Estimating the noise mask was a weak point in the paper cited above. Our goal has been to improve on the noise mask generation using machine learning methods.

Since the automatic speech recognition system used in the CHiME project (HTK) uses the Mel filterbank energies as features for a hidden Markov model, we started by using these same filterbank energies as features for the support vector machine to estimate a noise mask. We have used the freely available tools LIBLINEAR [2] and LIBSVM [3] for training and prediction rather than writing an SVM from scratch, allowing us to focus on feature selection, kernel selection, etc. Since the noise mask generates a label of reliable/unreliable for each frequency bin, time frame pair, we needed to train a separate SVM for each frequency bin.

The full data set consisted of hundreds of thousands of training examples (where each training example consists of a time frame of Mel filterbank energies for each frequency bin). This large data set had the potential to result in long training and prediction times. Twenty separate SVMs needed to be trained, one for each frequency bin of interest. We experimented and found that the training accuracies for the SVMs for different frequency bins were relatively similar, so to keep the problem tractable, we focused on a single frequency bin (one SVM) while learning about the SVM.

We started by using the log filterbank energies for all frequency bins in a given time frame as features for all SVMs (all 20 SVMs had the same features, just different labels). We used a linear kernel (LIBLINEAR) and found that the classification accuracy was unacceptably low. Using the middle frequency bin (k=10) and 100,000 examples (out of roughly 600,000 total training examples, split with 90,000 training examples and 10,000 test examples), the linear kernel resulted in a classification accuracy of 56.3%. Scaling the features to a range of

[-1,1] improved the accuracy to 64.7% while reducing the runtime by an order of magnitude.

Since the training set was quite large, training and prediction with a nonlinear kernel (using LIBSVM) was very slow, so we pursued improving the accuracy of the linear kernel. By observing the training and testing accuracy for various training set sizes, we determined that the SVM using a linear kernel was underfitting the data, so we considered what additional features could be added that would be relevant to generating an accurate noise mask.

Temporal information about what the spectrum of the audio is before and after the time frame of interest could be used to provide additional information to the classifier. Adding features for the frame before and the frame after for all frequency bins improved the accuracy relative to only scaling the features from 64.7% to 65.8% for our test setup (90,000 training examples, 10,000 test examples, frequency bin 10, linear kernel).

We thought that there was likely to be some correlation between the features that was not accounted for with the linear kernel, so we investigated adding features that consisted of product terms of the features. This improved accuracy relative to only scaling the features from 64.7% to 70.1%. However, adding the squared features meant that the number of features went from 20 to 400. We were limited by the virtual address space available in Matlab (32-bit student edition) in addition to the runtime for SVM training and classification, so we were limited in the number of additional features we could add.

In the CHiME setup, the speaker was in a fixed location while the noise could be at any location, and the audio was recorded via two microphones. Up to this point we had been using the average of the two audio channels to find the log filterbank energies to be used as features. We added additional log filterbank energy features for the difference between the audio channels. Using scaling, difference and repeated time frames improved the accuracy slightly from 70.1% to 70.5%.

At this point our classification accuracy on the training set was still unacceptably low and the linear kernel was still underfitting the data. We were running into issues adding additional features because we were hitting the maximum array size in 32-bit Matlab. Despite the high runtime of the SVM using a nonlinear kernel (libSVM) we knew we likely needed a nonlinear kernel in order to have the model complexity necessary to fit the data.

Similar to the linear kernel, we recorded accuracy vs model complexity for several possible feature sets. Using the original unscaled log filterbank energy features, a radial basis function kernel, the same 100,000 training examples but split using 4-way cross validation (vs 90/10 without cross-validation for the linear kernel), the SVM achieved an accuracy of 52.3% on the middle frequency bin (k=10). This is similar to what the linear kernel achieved. We performed a search over the SVM parameters, using cross-validation to select the optimal parameters (unfortunately we did not record a before/after accuracy for this step). Using scaled features increased the accuracy to 71.9% while decreasing the runtime. Adding the difference features that were previously described along with scaling increased the accuracy from 71.9% to 77.4% Adding the features for the time frame before and after the frame of interest along with scaling and using difference features increased the cross-validation accuracy from 77.4% to 82.1%. At this point the accuracy when testing on the same data as the SVM was trained resulted in an accuracy of ~87% vs 82.1% using cross-validation, so the test accuracy was approaching the training accuracy. It was difficult to tell whether the training accuracy would converge to the test accuracy given enough data or not. However we knew that the training accuracy represented an upper bound on the accuracy for the current feature set. We added one more set of features representing the speaker identify

(which is known for the CHiME dataset), where a '1' in a particular position in the feature vector indicates that that speaker is talking. After proper scaling and removal of empty rows in the feature matrix, the test accuracy using the same 100,000 training example subset improved from 82.1% to 84.6%.

With this feature set we began looking into using the full dataset as well as feature selection and parameter selection. Using scaled features, difference features, the optimal parameters found for a smaller dataset, and all 626,000 training examples (with 90% used for training and 10% used for testing), the SVM achieved a classification accuracy of 82.3%, similar to the test accuracy of 82.1% with the same features and a 100,000 training example subset of the data. Adding the speaker identity features previously described improved the test accuracy of 86.4%, indicating that the increased model complexity that resulted from adding the speaker identity features increased the variance when using a subset of the data.

Training and testing with the full dataset required several days of runtime. Feature selection and parameter selection both require cross-validation for multiple permutations of parameters and features. We initially performed feature selection and parameter selection using a 10,000 training example subset of the dataset. Cross-validation accuracy degraded when using a subset of the features, so feature selection using this subset of the data did not indicate that it was possible to reduce the feature set. We also found the optimal features through cross-validation on the 10,000 example subset of the data. However, intuition and experimentation indicated that the optimal parameters were not constant across training set size, since the model parameters impacted the bias versus variance tradeoff. We chose to perform feature selection and parameter selection again using a larger subset of the data (160,000 training examples) and 5-way cross-validation for each setting (Figure 2). The feature selection indicated that reducing the feature set size would degrade the SVM classification accuracy significantly so we did not pursue this further. As expected, the optimal parameters for the larger training set size were not the same as for a smaller training set size. We retrained the SVM with the new estimated optimal parameters, but unfortunately the SVM training and prediction took several days and did not quite complete in time. In parallel, we used the same full dataset (626,000 examples, 90% training, 10% testing) with the new estimated optimal parameters to correlate with another frequency bin. We chose bin k=3 as a bin with a high concentration of speech energy. This run did complete in time, and we found a classification accuracy on the test set of 89.7%. This concluded our optimization of the SVM. Given the time required for training and testing the SVM with the full dataset, we did not extend training and testing to SVMs for other frequency bins apart from the correlation run described above, although our experimentation with smaller dataset sizes indicates that the training and testing accuracy is similar for other frequencies.

Figure 2 below shows the oracle noise mask, the SVM estimated noise mask, and the difference between the masks for a subset of the test dataset. For runtime reasons, a subset of the training dataset was used to train the SVM when generating this plot, so the training accuracy was lower (~75%) than what the SVM achieved for the full training and test datasets. Table 1 summarizes the classification accuracy versus settings using a non-linear kernel in libSVM.

In this paper we have described the design of an SVM classifier which takes audio clips and generates a noise mask with labels of reliable or unreliable for time frame and frequency bin pairs. The noise mask can be used by a data imputation system to replace unreliable audio with an estimate of the clean audio to improve automated speech recognition accuracy. The features used by the SVM are the log Mel filterbank energies for the average and difference of the audio channels along with the speaker identity. The resulting SVM achieved a

classification accuracy of 89.7% for a speech-dominated frequency bin, which is representative of the accuracy that can be achieved for the SVMs for other frequencies.

Table 1. Comparison of classification accuracy vs feature set and settings

Features, setup	**Accuracy**	**Notes**
Unscaled FBE, libsvm	52.3%	100k examples, 4-way CV, freq k=10
Scaled FBE, libsvm	71.9%	Same as above
Scaled FBE, difference features, libsvm	77.4%	Same as above
Scaled FBE, difference features, before/after features, libsvm	82.1%	Same as above
Scaled FBE, difference features, before/after features, , speaker identity, libsvm	84.6%	100k examples, 90% train, 10% test, k=10
Same as above	86.4%	626k examples, 90% train, 10% test, k=10
Same as above with optimized SVM parameters	Didn't finish in time	Same as above
Same as above with optimized SVM parameters	89.7%	Same examples as above, freq k=3

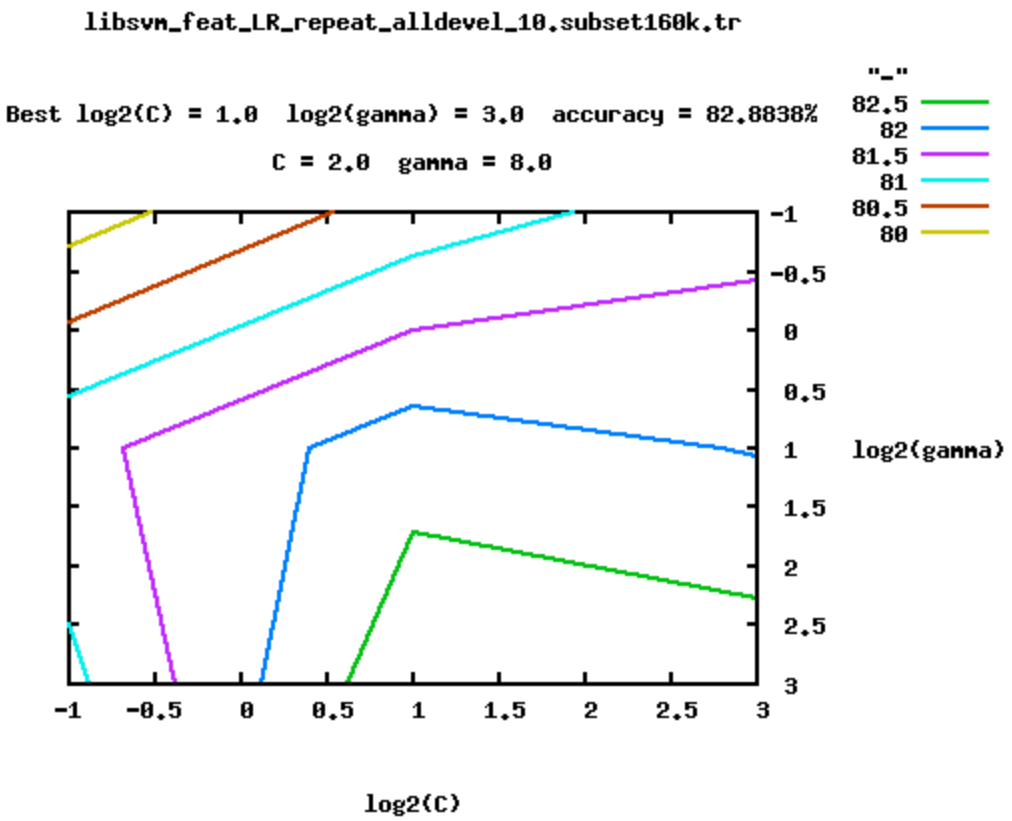

Figure 1. Classification accuracy versus SVM parameters using 160,000 examples

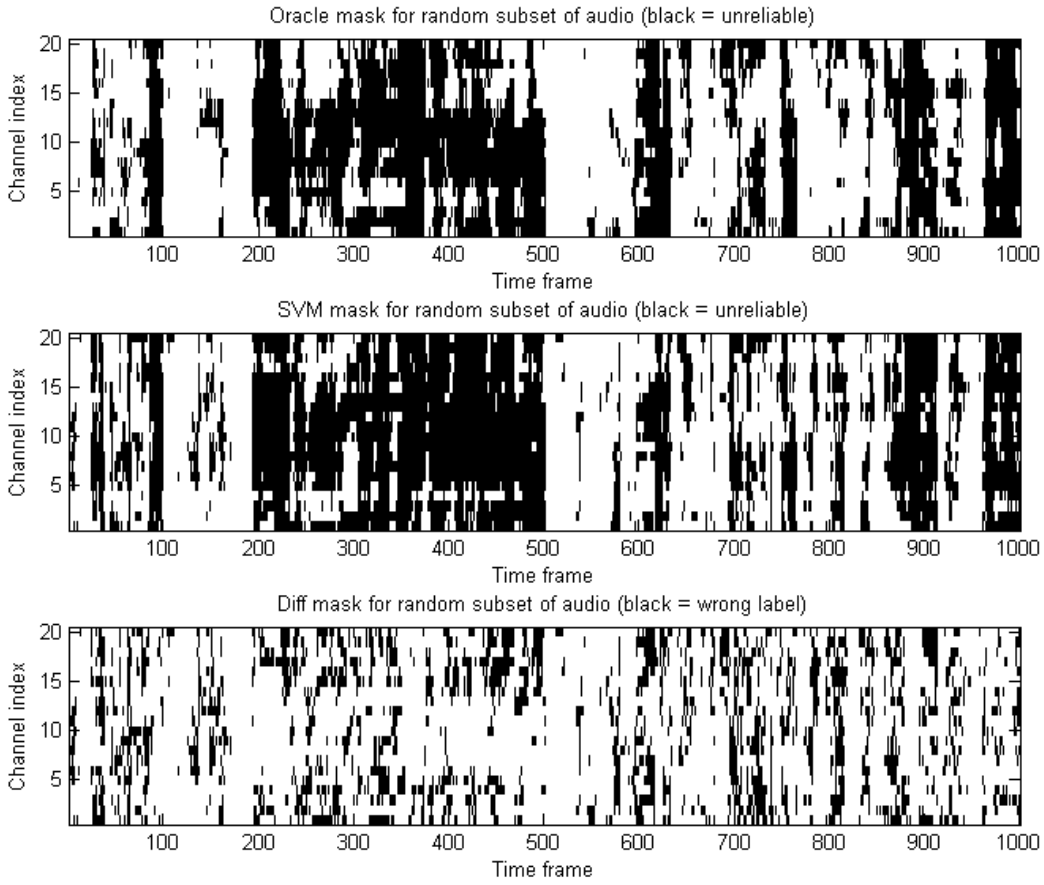

Figure 2. SVM estimated noise mask versus oracle mask

References

[1] Kallasjoki, Keronen, Gemmmeke, Remes, Palomaki, "Mask estimation and sparse imputation for missing data speech recognition in multisource reverberant environments," in *CHiME 2011 Workshop on Machine Listening in Multisource Environments*

[2] LIBLINEAR -- A Library for Large Linear Classification. Machine Learning Group at National Taiwan University. 2012. 8 Dec. 2012 http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[3] LIBSVM -- A Library for Support Vector Machines. Chih-Chung Chang and Chih-Jeh Lin. 2012. 8 Dec. 2012 http://www.csie.ntu.edu.tw/~cjlin/libsvm/