Temporal Feature Selection for Noisy Speech Recognition

Ludovic Trottier, Brahim Chaib-draa, and Philippe Giguère

Department of Computer Science and Software Engineering,
Université Laval, Québec (QC), G1V 0A6, Canada
ludovic.trottier.1@ulaval.ca
{chaib,philippe.giguere}@ift.ulaval.ca
http://www.damas.ift.ulaval.ca

Abstract. Automatic speech recognition systems rely on feature extraction techniques to improve their performance. Static features obtained from each frame are usually enhanced with dynamical components using derivative operations (delta features). However, the susceptibility to noise of the derivative impacts on the accuracy of the recognition in noisy environments. We propose an alternative to the delta features by selecting coefficients from adjacent frames based on frequency. We noticed that consecutive samples were highly correlated at low frequency and more representative dynamics could be incorporated by looking farther away in time. The strategy we developed to perform this frequency-based selection was evaluated on the Aurora 2 continuous-digits and connected-digits tasks using MFCC, PLPCC and LPCC standard features. The results of our experimentations show that our strategy achieved an average relative improvement of 32.10% in accuracy, with most gains in very noisy environments where the traditional delta features have low recognition rates.

Keywords: automatic speech recognition, delta features, feature extraction, noise robustness.

1 Introduction

Automatic speech recognition (ASR) is the transcription of spoken utterances into text. A system that performs ASR tasks takes an audio signal as input and classifies it into a series of words. In order to improve the performance of the system, feature extraction approaches are applied on the signal to provide reliable features. The three most frequently used features in ASR are the Mel frequency cepstral coefficients (MFCC), the perceptual linear predictive cepstral coefficients (PLPCC) and the linear predictive cepstral coefficients (LPCC) (see [18] for a review). These filter bank analysis extraction methods use various transformations, such as the Fourier transform, to convert a signal into a series of static vectors called *feature frames*. The coefficients in a feature frame are usually ordered from low-frequency to high-frequency and this observation will play a central role in our approach.

Classical feature extraction methods enhance each feature frame with dynamical components by concatenating the first- and second-order derivatives. These *delta* features were proposed as a way to improve the spectral dynamics of static features [3]. Delta features improve the accuracy of the hidden Markov model (HMM) [23] by reducing the impacts of the state conditional independence [4]. However, it is known from signal processing theories that the derivative of a noisy signal amplifies the noise and reduces the quality of the extracted information [14]. This can be especially harmful in real world situations where noise affects the recognition, such as when driving a car [13].

We have proposed, in a preliminary approach, that the delta features could be replaced with a mere concatenation of adjacent (in time) coefficients based on frequency [19]. This approach will be referred to as Temporal Feature Selection (TFS). The suggestion that dynamical features should be dependent on frequency was motivated by the importance of modeling inter-frame dependencies for speech utterances. Signal processing theories suggest that information in a signal varies according to its frequency [14]. For example, implosive consonant will result in fast, high-frequency features, while vowels will produce slow-changing, lower-frequency features. It thus appears that dynamical features may be enhanced by measuring the variation of the signal's information with frequency.

In this paper, we extend our TFS method with a learning framework. Our framework uses the variance of the difference of adjacent feature frames as a way to identify the positions where more reliable dynamical information resides. We show experimentally that our dynamical features improve the accuracy over the classical delta features on the Aurora 2 [15] continuous-digits and connected-digits tasks.

The rest of the paper is organized as follows. Section 2 describes related approaches, section 3 contains background information about feature extraction, section 4 presents the TFS method, section 5 details the experimentations and section 6 concludes this work.

2 Related Work

To overcome the delta features' problem of susceptibility to noise, recent alternatives have been investigated. The delta-spectral cepstral coefficients (DSCC) have been proposed in replacement of the delta features to add robustness to additive noise [10]. Also, the discrete cosine transform (DCT) has been used in a distributed fashion (DDCT) prior to the calculation of the delta features [8]. Finally, a weighted sum combining the static and delta features have been proposed in replacement of the usual concatenation [20]. The main drawback of all these methods is that derivative operations are still part of their processing pipeline thus making the features prone to be corrupted by noise.

Additionally, splicing followed by decorrelation and dimensionality reduction has been used to enhance the inputs of deep neural networks (DNNs) [16]. Splicing consists in concatenating all feature frames (with delta features) in a *context*

window of size c around each frame [1]. Moreover, it was showed that deeper layers allow more discriminative and invariant features to be learned [22]. While we acknowledge that deep learning is a promising avenue for feature extraction in ASR, we argue that better feature engineering methods could facilitate the DNN learning process.

In the context of linear feature transformations, some have looked at dimensionality reduction approaches such as Principal Component Analysis (PCA) [9], Linear Discriminant Analysis (LDA) [2], Heteroscedastic LDA (HLDA) [11] amd Heteroscedastic DA (HDA) [17]. These approaches are essential tools for speech feature extraction, but we argue that they may be avoided by using a better model for gathering the speech dynamics. Linear transformations of speech features have also been applied for decorrelation, such as Maximum Likelihood Linear Transform (MLLT) [7], Global Semi-tied Covariance (GSC) [6], and for speaker adaptation, such as feature-space Maximum Likelihood Linear Regression (fMLLR) [12] and Constrained MLLR (CMLLR) [5]. However, these feature selection techniques do not address the problem of modeling speech dynamics.

3 Background

In this section, we review in brief the steps for performing MFCC feature extraction on speech signals. The overview of the method is presented in Fig. 1. For additional details on this technique and other related approaches (such as PLPCC and LPCC), see [18].

Pre-emphasis: A speech waveform entering the pipeline is first filtered with a first order high pass filter. The goal of this transformation is to remove the low-frequency parts of the speech, as they tend to have similar and redundant adjacent values.

Windowing: The resulting signal is them divided into 20-40 milliseconds *frames*. A length of 25 ms is typical in speech processing. Assuming the signal is sampled with a frequency of 8 kHz, the frame length corresponds to $0.025 * 8000 = 200$ samples. Usually, the frames are overlapping by 15 ms (120 samples at 8 kHz), which means that a frame is extracted at every 10 ms (80 samples at 8 kHz) in the signal.

Periodogram Estimate of Power Spectrum: To perform the periodogram estimate of the power spectrum, the discrete Fourier transform (DFT) is applied on each frame to transform the waveform into its frequency domain. DFT assumes that each signal is periodic, which means that the beginning and the end of each frame should be connected. For a randomly selected frame, this hypothesis will not be respected and will lead to abrupt transitions. The discontinuities at the edges will be reflected in the spectrum by the presence of spectral leakage. To get a better resolution, the Hamming windowing function is applied to connect the edges in a smoother way. The length of the DFT is typically 512, but only the first 257 coefficients are kept since the other 255 are redundant due to the nature of the Fourier transform. Finally, the squared absolute value of the DFT is applied which gives the periodogram estimate of power spectrum.

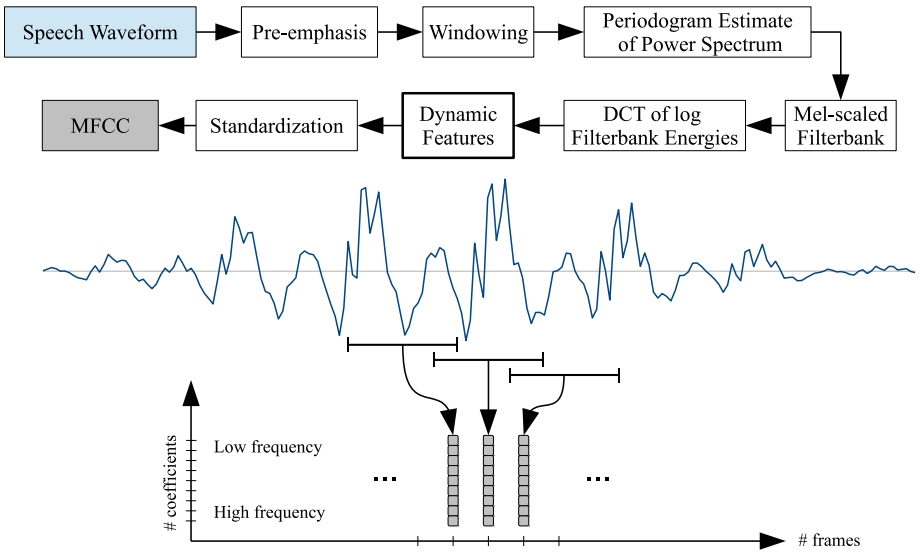


Fig. 1. MFCC extraction.

Mel-scaled Filterbank: Each power spectral estimate is filtered using triangular Mel-spaced filterbanks (see [21] for more details). The filterbanks are described as 26 vectors of size 257 (assuming $K = 512$). Each vector contains mostly zeros excepted at certain regions of the spectrum and thus act as band-pass filters. Mapping the frequency to the Mel scale allows the features to match more closely the non linear perception of pitch of the human auditory system. To compute the filterbank energies, we can simply multiply the periodogram estimates by each filterbank and sum the values. The 26 numbers give an indication of the amount of energy in each filterbank.

DCT of log Filterbank Energies: Then, a type 2 discrete cosine transform (DCT-II) is performed on the log filterbank energies to decorrelate the values. The length of the DCT is usually 14 and the first coefficient is discarded. The resulting 13 coefficient correspond to the static features.

Dynamic Features: As explain in section 1, the impacts of the state conditional independence of the HMM can be reduced by gathering dynamical information. In most cases, the delta features are appended to the static features by computing discrete derivatives (more details in section 4.2).

Standardization: Finally, we subtract each coefficient with its sample mean and divide by its sample standard deviation. These statistics can be calculated once for all utterances or individually for each utterance.

As shown in Fig. 3, the signal is transformed into a series of vectors, one for each frame extracted during windowing. Each utterance has a different number of frames, depending on its length and its sampling frequency. We now present

our approach that is an alternative method to the computation of delta features during the *Dynamic Features* step.

4 Temporal Feature Selection

4.1 Definition

Let $\Phi^{(n)} = \left(\boldsymbol{\phi}_{:,1}^{(n)} \dots \boldsymbol{\phi}_{:,T_n}^{(n)}\right)$, $n = 1 \dots N$, be a $D \times T_n$ matrix of D-dimensional static features. N is the total amount of utterances and T_n denotes the number of frames extracted from utterance n. For example, $\Phi^{(n)}$ could represent MFCC, as presented in section 3. We denote the column vector $\boldsymbol{\phi}_{:,t}^{(n)}$ as the feature frame at position t. The classical method of computing the delta features uses the following equations:

$$\Delta\boldsymbol{\phi}_{:,t}^{(n)} = \frac{\sum_{k=1}^{K} k \left(\boldsymbol{\phi}_{:,t+k}^{(n)} - \boldsymbol{\phi}_{:,t-k}^{(n)}\right)}{2\sum_{k=1}^{K} k^2}, \tag{1}$$

$$\Delta\Delta\boldsymbol{\phi}_{:,t}^{(n)} = \frac{\sum_{k=1}^{K} k \left(\Delta\boldsymbol{\phi}_{:,t+k}^{(n)} - \Delta\boldsymbol{\phi}_{:,t-k}^{(n)}\right)}{2\sum_{k=1}^{K} k^2}, \tag{2}$$

where $K = 2$ is a typical value for the summation. Although relevant dynamical information can be extracted with Eq. 1 and 2, the use of subtractions makes Δ and $\Delta\Delta$ features susceptible to noise.

The TFS features are, in contrast, coefficients taken from adjacent feature frames based on the frame position offsets $\mathbf{z} = [z_1, \dots, z_D]$. We define them as:

$$\tau\boldsymbol{\phi}_{i,t}^{(n)} = \left(\boldsymbol{\phi}_{i,t+z_i}^{(n)}, \boldsymbol{\phi}_{i,t-z_i}^{(n)}\right), \tag{3}$$

where z_i is a strictly positive integer. The parametrization of $\mathbf{z}$ is essential to extract robust dynamics. By imposing frequency dependency, τ will be constituted of coefficients $\boldsymbol{\phi}$ that are dissimilar, but not too much. The intuition is that too similar values do not increase the amount of information the feature frames carry, but increase its dimensionality, and this makes the speech recognition task harder. On the other hand, if the coefficients are too far apart, then their temporal correlation is meaningless.

4.2 Learning the TFS Features

We now present the proposed framework to learn the frequency dependent offsets $\mathbf{z}$. The method first computes the sample variance of the difference of neighboring feature frames. In other words, for each position t and utterance n, the difference between the feature frame $\boldsymbol{\phi}_{:,t}^{(n)}$ and its corresponding j^{th} neighbor $\boldsymbol{\phi}_{:,t+j}^{(n)}$ is computed. The variance of these differences is then calculated for $j \in \{1 \dots M\}$,

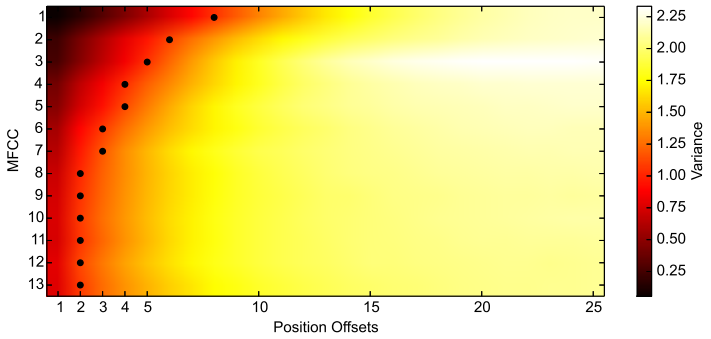


Fig. 2. Variance of the difference between a frame and its neighbors for MFCC features on the Aurora 2 [15] training dataset (best seen in colors). The coefficients are ordered from low frequency (1) to high frequency (13) for visual convenience (the proposed method does not require a specific ordering). The color refers to the variance of the difference Σ^M, where M was limited to 25 to reduce the computational burden.

where $M = \min\{T_1 \dots T_N\} - 1$. We define the matrix containing those values as:

$$\Sigma^M = \begin{bmatrix} \mathrm{Var}\left(\boldsymbol{\phi}_{1,t}^{(n)} - \boldsymbol{\phi}_{1,t+1}^{(n)}\right) & \dots & \mathrm{Var}\left(\boldsymbol{\phi}_{1,t}^{(n)} - \boldsymbol{\phi}_{1,t+M}^{(n)}\right) \\ \vdots & & \vdots \\ \mathrm{Var}\left(\boldsymbol{\phi}_{D,t}^{(n)} - \boldsymbol{\phi}_{D,t+1}^{(n)}\right) & \dots & \mathrm{Var}\left(\boldsymbol{\phi}_{D,t}^{(n)} - \boldsymbol{\phi}_{D,t+M}^{(n)}\right) \end{bmatrix}, \tag{4}$$

where the variances are taken over all positions t and utterances n. The variance is then be computed as follows:

$$\Sigma_{i,j}^M = \frac{1}{N_j^+} \sum_{n=1}^{N} \sum_{t=1}^{T_n - j} \left(\boldsymbol{\phi}_{i,t}^{(n)} - \boldsymbol{\phi}_{i,t+j}^{(n)} - \mu_{i,j}\right)^2, \tag{5}$$

where $\mu_{i,j}$ corresponds to the mean of the difference:

$$\mu_{i,j} = \frac{1}{N_j^+} \sum_{n=1}^{N} \sum_{t=1}^{T_n - j} \left(\boldsymbol{\phi}_{i,t}^{(n)} - \boldsymbol{\phi}_{i,t+j}^{(n)}\right), \tag{6}$$

and N_j^+ is the total number of frames:

$$N_j^+ = \sum_{n=1}^{N} T_n - j\,. \tag{7}$$

The purpose of computing Σ^M is to find the frame position offsets $\mathbf{z}$. Using the parameter V_{thresh} as a variance threshold, $\mathbf{z}$ is computed using the following

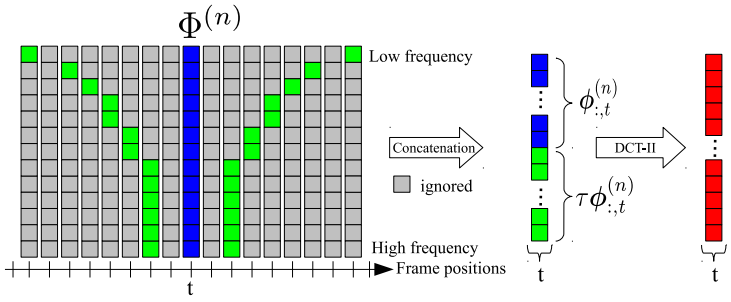


Fig. 3. Pipeline of processing for TFS features. After computing the frame position offsets $\mathbf{z}$ using Eq. 8, the features are concatenated and decorrelated with DCT-II.

equation:

$$z_i = \arg\min_j \left| \Sigma^M_{i,j} - V_{thresh} \right| , \tag{8}$$

where V_{thresh} is a hyper-parameter to choose. The frame position offsets $\mathbf{z}$ represented in Fig. 2 by the black dots are based on Eq. 8 for $V_{thresh} = 1$ and $M = 25$. In this particular example, $\mathbf{z} = [8, 6, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2, 2]$. For $i = 1$, this implies that $\tau\boldsymbol{\phi}_{1,t} = (\boldsymbol{\phi}_{1,t+8}, \boldsymbol{\phi}_{1,t-8})$.

What can be seen from this figure is that $\mathbf{z}$ depends on frequency. High frequency components have small offsets whereas low frequency components have large offsets. As explained in section 1, more reliable dynamical informations can be extracted from neighboring feature frames when frequency is taken into account. The relevant dynamical information of high frequency coefficients can only be extracted from nearly adjacent frames ($z_{13} = 2$). On the other hand, adjacent low frequency coefficients share most of their information and more time is needed to gather the relevant dynamics ($z_1 = 8$). Therefore, by using the variance of neighboring feature frames, $\mathbf{z}$ now incorporate the wanted characteristic of frequency dependency.

As in standard feature extraction, the features from each frame, $\boldsymbol{\phi}^{(n)}_{:,t}$ and $\tau\boldsymbol{\phi}^{(n)}_{:,t}$ are concatenated into a single column vector as shown in Fig. 3. Each resulting vector is then decorrelated with a type 2 DCT in order to accommodate it to the independence hypothesis of the Gaussian Mixture Model Hidden Markov Model (GMM-HMM) that was chosen as the inference method [4].

5 Experimental Results

5.1 Experimental Setup

The database that we used for our experiments is Aurora 2 [15] which contains a vocabulary of 11 spoken digits (*zero* to *nine* with *oh*). The digits are connected, thus they can be spoken in any order and in any amount (up to 7) with possible

pauses between them. The training set contains 8,440 utterances, both test set A and B have 28,028 and test C has 14,014 utterances. The utterances are noisy and the signal-to-noise ratio (SNR) varies from -5 dB, 0 dB, ..., 20 dB, Inf dB (clean). Different kinds of noise are present such as train, airport, car, restaurant, etc. On average, an utterance lasts approximately 2 seconds.

Using the HTK [21] framework provided with the Aurora 2 database, we performed two experiments. In the first one, 18 states whole-word HMMs were trained with a 3 components GMM (with diagonal covariance) as the state emission density. There was a total of 11 HMMs (one per class). In the second one, the whole-word HMMs were replaced with 5 states phoneme HMMs. In other words, using the CMU pronouncing dictionary, each digit was mapped to its ARPAbet interpretation. There was a total of 19 HMMs (one per phoneme). In our experimentations, we compared TFS (-T) to delta (-D) and double delta (-A) dynamic features on MFCC, PLPCC and LPCC. For all these features, 13 coefficients, including the energy (-E), excluding the 0th coefficient, were extracted to be used as observations. For all experiments, a variance threshold $V_{thresh} = 1$ was used. The performance of each method was averaged over all test sets for each noise level separately.

5.2 Experimental Results

The performances in word accuracy of our method are reported in Table 1 and 2 for whole-word and phoneme HMM respectively. In each table, the 7 noise levels from the Aurora 2 database are ordered from clean signals (SNR Inf) to highly noisy signals (SNR -5). The average over all noise levels is reported on the right. The last column consists of the relative improvement of the method over the reference model.

Based on these results, our approach achieved an averaged relative improvement of 20.79% for whole-word HMM and 32.10% for phoneme HMM. Also, it can be observed that TFS features increased the accuracy on all noisy tasks, but did not improve the results for clean signals with whole-word HMM. Nonetheless, these results support our initial intuition that using a pure derivative approach leads to inferior performances.

The variation of the word accuracy of TFS, with respect to V_{thresh}, is shown in Fig. 4 for whole-word HMMs. The performance of the method is reported for the 7 noise levels of the database. The crosses indicate the best result the method achieved for each noise level. This figure demonstrates the behavior of the performance of our approach with respect to the parametrization of $\mathbf{z}$.

5.3 Discussion

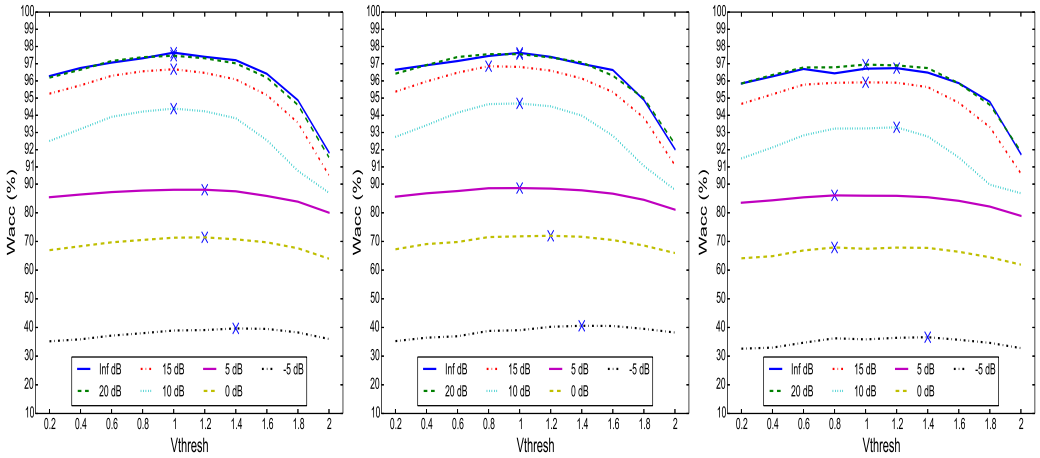
The results of table 1 show that the proposed TFS method does not outperform the delta features for clean utterances. This limitation is consistent with the intuition given in section 1 that dynamical components extracted using derivatives on clean utterances are not affected by amplified noise. However, this appears to be the case only for whole-word HMMs. As suggested by the results of table 2,

SNR (dB) / Features	Inf	20	15	10	5	0	-5	Avg.	R.I. (%)
(a) MFCC Based									
MFCC-E-D-A	98.54	97.14	96.02	93.27	84.86	57.47	23.35	78.66	-
MFCC-E-T $\mathbf{z} = [8, 6, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2, 2]$	97.64	97.46	96.68	94.39	88.03	71.31	38.93	**83.49**	**22.63**
(b) PLPCC Based									
PLPCC-E-D-A	98.65	97.56	96.48	93.85	85.93	59.83	25.36	79.66	-
PLPCC-E-T $\mathbf{z} = [8, 6, 5, 4, 4, 3, 3, 3, 3, 3, 2, 2, 2]$	97.40	97.36	96.60	94.52	88.43	71.98	40.23	**83.79**	**20.30**
(c) LPCC Based									
LPCC-E-D-A	98.30	96.82	95.59	92.28	81.87	54.52	22.96	77.48	-
LPCC-E-T $\mathbf{z} = [8, 6, 5, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2]$	96.74	96.90	95.90	93.30	85.91	67.86	36.39	**81.86**	**19.45**

Table 1. Word accuracy (%) of TFS (-E-T) features on the Aurora 2 database using whole-word HMMs. The results are averaged according to the noise level. The reference models are suffixed with -E-D-A.

SNR (dB) / Features	Inf	20	15	10	5	0	-5	Avg.	R.I. (%)
(a) MFCC Based									
MFCC-E-D-A	89.89	87.24	84.41	78.87	63.78	29.86	-5.82	61.17	-
MFCC-E-T $\mathbf{z} = [8, 6, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2, 2]$	93.02	94.15	92.65	88.84	79.22	56.42	19.58	**74.840**	**35.20**
(b) PLPCC Based									
PLPCC-E-D-A	88.99	87.92	84.78	78.97	64.18	32.96	-0.67	62.45	-
PLPCC-E-T $\mathbf{z} = [8, 6, 5, 4, 4, 3, 3, 3, 3, 3, 2, 2, 2]$	92.98	94.29	92.89	89.38	79.76	56.86	18.79	**74.99**	**33.40**
(c) LPCC Based									
LPCC-E-D-A	88.13	86.04	83.11	76.64	60.82	29.65	0.02	60.63	-
LPCC-E-T $\mathbf{z} = [8, 6, 5, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2]$	91.09	93.26	91.24	86.98	76.45	50.90	10.79	**71.53**	**27.69**

Table 2. Word accuracy (%) of TFS (-E-T) features on the Aurora 2 database using phoneme HMMs. The results are averaged according to the noise level. The reference models are suffixed with -E-D-A.



(a) Using MFCC features. (b) Using PLPCC features. (c) Using LPCC features.

Fig. 4. Variation of the performance of TFS as a function of V_{thresh} using whole-word HMM. The crosses indicate the maximum word accuracy for each noise level.

the TFS method improves the word accuracy even in the absence of noise when using phoneme HMMs. These non intuitive results may be related to the shorter duration of phonemes in comparison to whole-words. By selecting very distanced coefficients, TFS also incorporates information about adjacent phonemes and appears to simulate triphone modeling, where a HMM is define for every phoneme triplets. This was not the case with whole-word HMMs because the state occupancy of a phoneme HMM is usually much shorter.

It can be seen in Fig. 4 that the plots are approximatively convex. The up and down hill-shaped curves support the idea introduced in sections 1 and 4 relative to informative coefficients. The amount of unrelated information added to the frame will be greater than the amount of related information if the concatenated coefficients are too close, or too far, from each other. This phenomenon is reflected in Fig. 4 with an increase in accuracy when V_{thresh} increases, up to some point where it starts to decrease.

Another observation that is worth mentioning in Fig. 4 is the behavior of the best accuracy with respect to the noise level. Apart from LPCC, where it is less clearly identifiable, the maximum result tends to occur at greater V_{thresh} as the noise increases. For example, the best word accuracy appears at $V_{thresh} = 1$ for the least noisy task and at $V_{thresh} = 1.4$ for the noisiest one. Since a greater threshold produces a $\mathbf{z}$ that has greater time offsets, our approach seems to act like a noise reduction method by smoothing the signal. Indeed, smoothing a highly noisy signal requires gathering information at a far distance. In this sense, our approach behaves similarly by selecting coefficients that are farther apart.

In summary, our results suggest that frequency-based dynamical features relying on the concatenation of adjacent coefficients helps improve the accuracy, especially for noisy utterances. The results on the Aurora 2 database show that the proposed TFS features achieved a better average word accuracy than the delta features. However, in the context of recognizing clean utterances with whole-word HMMs, our method did not outperform the reference features. Nonetheless, TFS appears to be a good choice for dynamical features since it performed the best overall, can be learned rapidly from the data and is based on a single specified parameter V_{thresh}.

6 Conclusion

A novel way of improving the dynamics of static speech features was proposed. The issue that was addressed was the susceptibility to noise of derivative operations during the modeling of speech dynamics. The proposed Temporal Feature Selection (TFS) features have shown to improve the robustness of the state of the art delta features in various types of noise. The experimentations have shown that the 3 most standard features, MFCC, PLPCC and LPCC, combined with the TFS features achieved an averaged relative improvement of 20.79% and 32.10% in accuracy for whole-word and phoneme HMMs on the Aurora 2 database.

For further study, we plan to evaluate our approach on the harder problem of large vocabulary continuous speech recognition. Specifically, we will examine to potential of TFS to replace triphone HMMs modeling. Finally, we intend to use deep learning approaches to study the impacts of better feature engineering on the learning process of DNNs.

References

1. Bahl, L., De Souza, P., Gopalakrishnan, P., Nahamoo, D., Picheny, M.: Robust methods for using context-dependent features and models in a continuous speech recognizer. In: Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on. vol. 1, pp. I–533. IEEE (1994)
2. Fukunaga, K.: Introduction to Statistical Pattern Recognition (2nd Ed.). Academic Press Professional, Inc., San Diego, CA, USA (1990)
3. Furui, S.: Speaker-independent isolated word recognition based on emphasized spectral dynamics. In: Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'86. vol. 11, pp. 1991–1994. IEEE (1986)
4. Gales, M., Young, S.: The application of hidden markov models in speech recognition. Foundations and Trends in Signal Processing 1(3), 195–304 (2008)
5. Gales, M.J.: Maximum likelihood linear transformations for hmm-based speech recognition. Computer speech & language 12(2), 75–98 (1998)
6. Gales, M.J.: Semi-tied covariance matrices for hidden markov models. Speech and Audio Processing, IEEE Transactions on 7(3), 272–281 (1999)
7. Gopinath, R.A.: Maximum likelihood modeling with gaussian distributions for classification. In: Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on. vol. 2, pp. 661–664. IEEE (1998)

8. Hossan, M.A., Memon, S., Gregory, M.A.: A novel approach for MFCC feature extraction. In: Signal Processing and Communication Systems (ICSPCS), 2010 4th International Conference on. pp. 1–5. IEEE (2010)
9. Jolliffe, I.: Principal component analysis. Springer Series in Statistics, Berlin: Springer, 1986 1 (1986)
10. Kumar, K., Kim, C., Stern, R.M.: Delta-spectral cepstral coefficients for robust speech recognition. In: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on. pp. 4784–4787. IEEE (2011)
11. Kumar, N., Andreou, A.G.: Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. Speech communication 26(4), 283–297 (1998)
12. Leggetter, C.J., Woodland, P.C.: Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. Computer Speech & Language 9(2), 171–185 (1995)
13. Lockwood, P., Boudy, J.: Experiments with a nonlinear spectral subtractor (NSS), Hidden Markov models and the projection, for robust speech recognition in cars. Speech Communication 11(2–3), 215 – 228 (1992)
14. Oppenheim, A.V., Schafer, R.W., Buck, J.R.: Discrete-time Signal Processing (2nd Ed.). Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1999)
15. Pearce, D., günter Hirsch, H., Gmbh, E.E.D.: The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In: ISCA ITRW ASR2000. pp. 29–32 (2000)
16. Rath, S.P., Povey, D., Veselỳ, K.: Improved feature processing for deep neural networks. In: Proc. Interspeech (2013)
17. Saon, G., Padmanabhan, M., Gopinath, R., Chen, S.: Maximum likelihood discriminant feature spaces. In: Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on. vol. 2, pp. II1129–II1132. IEEE (2000)
18. Shrawankar, U., Thakare, V.M.: Techniques for feature extraction in speech recognition system: A comparative study. arXiv:1305.1145 (2013)
19. Trottier, L., Chaib-draa, B., Giguère, P.: Effects of frequency-based inter-frame dependencies on automatic speech recognition. In: Canadian Conference on AI. pp. 357–362 (2014)
20. Weng, Z., Li, L., Guo, D.: Speaker recognition using weighted dynamic MFCC based on GMM. In: Anti-Counterfeiting Security and Identification in Communication (ASID), 2010 International Conference on. pp. 285–288. IEEE (2010)
21. Young, S.J., Evermann, G., Gales, M.J.F., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P.C.: The HTK Book, version 3.4. Cambridge University Engineering Department, Cambridge, UK (2006)
22. Yu, D., Seltzer, M.L., Li, J., Huang, J.T., Seide, F.: Feature learning in deep neural networks-studies on speech recognition tasks. arXiv:1301.3605 (2013)
23. Zheng, F., Zhang, G., Song, Z.: Comparison of different implementations of MFCC. Journal of Computer Science and Technology 16(6), 582–589 (2001)