Normalization of the Speech Modulation Spectra for Robust Speech Recognition

Xiong Xiao*, Student Member, IEEE, Eng Siong Chng, Senior Member, IEEE,
and Haizhou Li, Senior Member, IEEE

Abstract—In this paper, we study a novel technique that normalizes the modulation spectra of speech signals for robust speech recognition. The modulation spectra of a speech signal are the power spectral density (PSD) functions of the feature trajectories generated from the signal, hence they describe the temporal structure of the features. The modulation spectra are distorted when the speech signal is corrupted by noise. We propose the temporal structure normalization (TSN) filter to reduce the noise effects by normalizing the modulation spectra to reference spectra. The TSN filter is different from other feature normalization methods such as the histogram equalization (HEQ) that only normalize the probability distributions of the speech features. Our previous work showed promising results of TSN on a small vocabulary Aurora-2 task.

In this paper, we conduct an inquiry into the theoretical and practical issues of the TSN filter that includes: 1) We investigate the effects of noises on the speech modulation spectra and show the general characteristics of noisy speech modulation spectra. The observations help to further explain and justify the TSN filter. 2) We evaluate the TSN filter on the Aurora-4 task and demonstrate its effectiveness for a large vocabulary task. 3) We propose a segment-based implementation of the TSN filter that reduces the processing delay significantly without affecting the performance. Overall, the TSN filter produces significant improvements over the baseline systems, and delivers competitive results when compared to other state-of-the-art temporal filters.

Index Terms: robust speech recognition, modulation spectrum, feature normalization, temporal filter, Aurora task

I. INTRODUCTION

ROBUST performance in noisy real life environments is one of the major challenges in speech recognition research [1, 2]. In real life applications, speech signals are usually affected by various factors such as ambient background noises, reverberation effects, speech signals from competing speakers, and unmatched microphone characteristics. Generally, these factors change the statistics of speech features such that the acoustic model trained using clean speech features no longer represents the statistics of distorted speech features. The mismatch between the features and the model inevitably results in degraded recognition performance. To improve the performance, current techniques focus on reducing the statistical mismatch between the clean-trained speech model, e.g. hidden Markov models (HMM), and the distorted speech features.

The feature-model mismatch can be reduced by two approaches, the feature compensation approach and the model adaptation approach. The feature compensation approach modifies the noisy features towards the unobserved clean features during or after the feature extraction process. Such techniques include the Wiener filter [3], the spectral subtraction method [4, 5], the minimum mean square error (MMSE) estimator of the speech spectral magnitude [6], the log filterbank coefficient estimators [7–9] and the cepstral coefficient estimator [10], etc. In contrast, the model adaptation approach adapts the clean-trained acoustic model to better represent the noisy features. Examples of this approach include the parallel model composition (PMC) [11], the maximum a posteriori adaptation (MAP) [12], and the maximum likelihood linear regression adaptation (MLLR) [13].

One major approach in feature compensation involves normalizing the statistics of speech features [14–19]. As the statistics of features are changed when the speech signal is corrupted by noise, the normalization of the feature statistics will reduce noise effects. Normalization techniques include cepstral mean normalization (CMN) [14], cepstral variance normalization (CVN) [15], and histogram equalization (HEQ) [16–19]. From the statistical viewpoint, the CMN, CVN and HEQ normalize the first order moments, the second order moments and the probability distributions of the speech features, respectively. A major advantage of the feature normalization techniques is that they are independent of noise statistics, which are difficult to estimate especially when the noise is non-stationary and/or similar to the speech signal, e.g. the babble noise. Hence, normalization techniques are simple and easy to implement and also produce significant improvements to recognition accuracy under noisy conditions [15–19].

Besides normalizing the probability distributions of speech features, the distortions of features in the temporal domain also need to be reduced for robust speech recognition. Temporal characteristics of the features are important for speech recognition. Successful examples of using temporal information include the HMM modeling of speech features and its derivatives [20–24], the temporal-derived features [25], the modulation spectrogram features [26], and the delta and acceleration features [27]. When speech is distorted by noise, the temporal characteristics of feature trajectories are also distorted and need to be enhanced.

Temporal filters [28–35] improve the temporal characteristics of speech features by modifying the PSD functions of the feature trajectories. The temporal filtering is usually applied to log filterbank coefficients or cepstral coefficients.
The power spectral density (PSD) functions of these coefficients are closely related to the modulation spectra of the speech signal which represents the power of amplitude modulation signals [36–38]. Past research showed that the modulation spectra correlate with speech intelligibility and some modulation frequencies are more important than others. For example, the 1-16Hz modulation frequency range of speech is found to be more important to speech intelligibility and automatic speech recognition than the very low (<1Hz) and high (>16Hz) modulation frequency ranges [36, 37, 39–42]. Motivated by this, temporal filters are usually designed to enhance the useful modulation frequency range for speech recognition and attenuate other modulation frequency ranges. For instance, the relative spectra (RASTA) filter [28] is an infinite impulse response (IIR) filter whose passband is about 1-12Hz. The autoregressive moving average (ARMA) filter in the MVA technique [29] is a low-pass filter and its cut-off frequency depends on the filter length. Other researchers have also proposed to design filters from the speech data to better represent the features or improve the features’ discriminative ability [30–35]. Data-driven filters are usually band-pass like RASTA, or low-pass like MVA. Considerable improvements to recognition accuracy by these temporal filters have been reported in the literature [28–35].

Despite many promising results, both temporal filters and normalization techniques can be further improved. First, current temporal filters such as RASTA, ARMA and the datadriven filters are fixed filters and do not adapt to environment conditions. However, it is well understood that filtering of features should respond to environment changes. For example, it is reasonable to smooth very noisy features more aggressively than clean features. Second, current normalization techniques such as HEQ only normalize the probability distributions of the features. We believe that temporal characteristics of the features represented by the speech modulation spectra can also be normalized for better performance. To this end, we proposed the TSN filter in [43].

The TSN filter is a temporal filter designed to normalize the modulation spectra of a speech utterance. The filter so designed shows adaptive ability to different environmental conditions, such as the signal-to-noise ratio (SNR) level. Our previous work [43–45] indicated that the adaptive TSN filter produced good results on the small vocabulary Aurora-2 task.

In this paper, we extend our study into several aspects of the TSN filter to provide new insights. We examine the effects of noises on the speech modulation spectra under different SNR levels. This study helps to explain the TSN filter’s magnitude response characteristics. In addition, we evaluate the TSN filter on the large vocabulary Aurora-4 task and another speech feature, i.e. the perceptual linear predictive (PLP) features [46]. Furthermore, to reduce the delay introduced by the filtering process, we propose a segment-based implementation of the TSN filter. To demonstrate the effectiveness of our approach, we compare the TSN filter to other popular temporal filters such as RASTA filter [28] and ARMA filter of MVA [29] in our experiments.

The paper is organized as follows. Section II presents the effects of noises on the speech modulation spectra in different stages of the feature extraction process together with a mathematical analysis. Section III briefly describes the TSN framework and the issues to be considered. Section IV discusses the segment-based implementation scheme of the TSN filter. The experimental results and discussions are presented in section V, and finally, we conclude in section VI.

II. NOISE EFFECTS ON SPEECH MODULATION SPECTRA

When a speech signal is corrupted by noises, its modulation spectra are also affected. To analyze the interactions between speech signal and noise, we first define the term “modulation spectra” used in this paper, and then discuss the characteristics of modulation spectra for speech signals and noises individually. By using a mathematical representation for the modulation spectra of noisy speech, we discuss how noises affect the characteristics of speech modulation spectra.

A. Definition of Modulation Spectra

The term modulation spectrum was introduced by Houtgast and Steeneken [36, 37] when they proposed the modulation transfer function (MTF) to measure speech intelligibility in the field of room acoustics. The meaning of modulation in speech signal is similar to that of amplitude modulation (AM) [48] in communication systems. In an AM system, a low frequency information-bearing signal is used to modulate the amplitude of a high frequency signal to transmit information through a medium. The speech signal can also be approximately viewed as an AM system: the very low frequency inaudible message-bearing waves are used to modulate the amplitudes of the audible sound waves [49]. Since speech is a wide band signal, it is more appropriate to analyze the amplitude modulation for individual frequency bands. In each frequency band, the speech signal can be considered as narrow band and its energy envelop can be seen as the modulating signal of the band. The modulating signal’s PSD function (the modulation spectrum) usually varies from band to band. The collection of the modulation spectra from all frequency bands forms the joint acoustic-modulation frequency representation of the speech signal [38]. The “acoustic frequency” here refers to the
“conventional Fourier decomposition of the signal” [38] and the “modulation frequency” is for the modulating signal.

In practice, the modulation spectra can be obtained as follows. For a time domain speech signal $x(i)$, its short-time power spectral density (i.e. the spectrogram, shown in the left panel of Fig. 1) is defined as:

$$|X(t, f)|^2 = |\text{STFT}[x(i)]|^2$$

(1)

where $i$, $t$ and $f$ are the sample index, frame index and the acoustic frequency index, respectively. The STFT hazards the short-time Fourier transform (STFT) and $|\cdot|$ denotes the magnitude. The trajectory of spectrogram coefficients in a single acoustic frequency bin represents the energy envelope of the speech signal in the acoustic frequency band centered at the bin (see the slices of the spectrogram in Fig. 1). Therefore, the trajectory can be interpreted as the down-sampled modulating signal (from signal sampling rate to frame rate of STFT) and its PSD function is the modulation spectrum for the acoustic frequency bin. The PSD functions for all the bins are the modulation spectra of the speech signal. While the maximum acoustic frequency is determined by the sampling frequency of the speech signal, the maximum modulation frequency is related to the frame rate of the STFT, which is the sampling rate of the feature trajectories. With a typical setup, the frame rate is 100Hz and the maximum modulation frequency is 50Hz.

In this paper, we loosely extend the definition of modulation spectra from spectrogram to feature coefficients, for example, to describe log filterbank coefficients and cepstral coefficients. In this way, the modulation spectra derived from feature coefficients are not the joint acoustic-modulation frequency representation of a speech signal, but rather the temporal characteristics of the feature coefficient trajectories. In the next section, we will examine the characteristics of modulation spectra generated from different coefficient trajectories.

B. Modulation Spectra of Clean Speech and Noises

The modulation spectra of speech signal are known to be affected by reverberation and additive noise. For example, reverberation usually shifts the peak of the modulation spectra from 4Hz to a lower modulation frequency due to the summation of signals from multiple transmission paths at the receiver [37]. When speech is corrupted by additive noise, it is intuitive that the modulation spectra will also be distorted. Besides reverberation and noise, the characteristics of modulation spectra may also depend on other factors, such as the spoken content of the utterance, the speaker’s characteristics, etc. In this paper, we are interested in examining environmental factors such as the corrupting noise and the SNR level that could affect the modulation spectra.

Before we examine the noisy speech’s modulation spectra (noisy modulation spectra), let’s first examine the modulation spectra of noises (noise modulation spectra) and the modulation spectra of clean speech (clean modulation spectra) separately. As speech and noise have very different acoustic characteristics, their modulation spectra will be different. We choose three kinds of noises from the Aurora-2 database [50] as examples: babble noise, car noise and subway noise. All these noises are recorded in typical real world environments.

The average modulation spectra of the three noises are shown in Fig. 2(a), together with the average modulation spectrum of clean speech. The average noise modulation spectra are estimated from 200 utterances (13 male speakers and 13 female speakers), and for each noise type, a 10 second long noise segment is used. The average clean modulation spectra are estimated from 200 utterances (13 male speakers and 13 female speakers), and the total length of these utterances is 340 seconds. A similar averaging method has also been used to analyze speech in [51] and to obtain smooth MTF in [52].

As the modulation spectra are averaged over 200 utterances, the effects of the spoken content and speaker are reduced and the observation shows the common trend of modulation

Fig. 2. The average modulation spectra of clean speech and three kinds of noises. The Yule-Walker method [47] is used to estimate the modulation spectra from feature coefficients. The order of the autoregressive (AR) model used in the Yule-Walker method is 6 for proper degree of details in the modulation spectra. The modulation spectra are represented in decibels where 0dB indicates a power of 1 and -10dB indicates 0.1. The modulation spectra are generated from the trajectories of: (a) the 8th Mel filterbank coefficients (the 8th Mel filterbank is centered at 656.25Hz with a bandwidth of about 280Hz); (b) the 8th log Mel filterbank coefficients; (c) the 8th cepstral coefficients. All the trajectories are processed by the mean and variance normalization (MVN) [15] to remove the DC offset and to normalize the power of the trajectories before the modulation spectra are estimated.
Our experimental studies have shown that the phase item is spectrum, the noise modulation spectrum and the phase item. The noisy modulation spectrum is the sum of the clean modulation spectra of clean speech and noises. From Fig. 2(a), it is clear that the modulation spectra of noises are flatter than that of speech. A similar observation has been reported in [41]. The observation is also true for modulation spectra generated from other filterbank trajectories, the log filterbank trajectories (see Fig. 2(b)) and the cepstral coefficient trajectories (see Fig. 2(c)).

An inquiry into the production process will help us understand the difference between the modulation spectra of human speech and noises. As the human speech production process is limited by the anatomical prerequisite of the human articulatory apparatus [53], the resulting sound patterns, their energy and changing rate in an utterance are bounded by the physical laws. As a result, in the modulation spectra, the power is concentrated in the low modulation frequencies. Unlike speech signals, background noises usually have less constraints and are more volatile. Therefore, the noises’ power spreads more evenly over the whole modulation spectrum from 0 to 50Hz. As we will show in the next section, the different characteristics between the speech and noise’s modulation spectra will determine the shape of the noisy modulation spectra.

C. Modulation Spectra of Noisy Speech

The modulation spectra of noisy speech signals are jointly determined by the clean speech signal and the corrupting noise. We derive the mathematical representation of the noisy modulation spectra generated from the filterbank coefficients in Appendix A. From the derivation, the noisy modulation spectrum of any filterbank is represented as

\[ P_y = P_x + P_n + Q \]  

where \( P_x \) and \( P_n \) are the modulation spectra of the clean speech and the corrupting noise for the same filterbank, respectively; \( Q \) represents the phase item in (14). From (2), the noisy modulation spectrum is the sum of the clean modulation spectrum, the noise modulation spectrum and the phase item. Our experimental studies have shown that the phase item is usually less significant than the other two items. In other words, the characteristics of the noisy modulation spectrum are mainly determined by the modulation spectra of clean speech and noise.

As speech and noise have different modulation spectra characteristics as shown in section II-B, the mixing of the speech and noise in (2) will inevitably change the shape of the noisy modulation spectra. In Fig. 3(a), the average noisy modulation spectra of the 8th filterbank coefficients for different SNR levels are shown. It is observed that the three noisy modulation spectra have higher power than the clean modulation spectrum in the high modulation frequency range, i.e. they are flatter than the clean modulation spectrum. In addition, the flatness of the noisy modulation spectra are inversely proportional to the SNR level.

These observations about the noisy modulation spectra are also true for log filterbank coefficients and cepstral coefficients. To generate the cepstral coefficients for speech recognition, the filterbank coefficients usually undergo two more processing stages: the dynamic range compression and the discrete cosine transform (DCT). The dynamic range compression reduces the dynamic range of the filterbank coefficients, e.g. logarithm compression [54], or \( n^{th} \) root compression [46]. The DCT reduces the feature vector size and decorrelates the feature dimensions so that the covariance matrix of cepstral features can be treated as diagonal [55]. Due to the complexity of these operations, it is difficult to analyze the modulation spectra derived from log filterbank coefficients and cepstral coefficients mathematically. In order to study the characteristics of the noisy modulation spectra in these two domains, we use experimental analysis. In Fig. 3(b),(c), the average noisy modulation spectra generated from log filterbank coefficients and cepstral coefficients are shown. From the figures, we find that the logarithm compression and the DCT process do not change the characteristics we observe for the filterbank coefficients. Like in the filterbank domain in Fig. 3(a), the noisy modulation spectra in the log filterbank and cepstral domains are generally flatter than the corresponding
clean modulation spectra. In addition, the flatness of the noisy modulation spectra are also inversely proportional to the SNR level.

D. Summary

In this section, we showed that the modulation spectra of noisy speech are different from those of clean speech due to noise corruption (see Fig. 3). In general, speech power concentrates in the low modulation frequencies more for the cleaner signal than for the noisier signal. This leads to different shapes of the modulation spectra demonstrated in Fig. 2-3. These observations show a need to correct the modulation spectra of noisy speech and we will discuss our method in the next section.

III. TEMPORAL STRUCTURE NORMALIZATION

The difference between clean and noisy speech signals in the modulation spectrum domain should be reduced to obtain robust speech features. To reduce the difference, we proposed the temporal structure normalization (TSN) filter [43]. For completeness, we will describe the TSN filter in this section.

A. Overview of TSN Framework

The TSN filter aims to reduce the temporal differences between the clean and noisy features, such as the smoothness of the feature trajectory. The magnitude response of the filter is designed from the PSD functions of the feature trajectories, i.e. the modulation spectra, since they represent the temporal characteristics of the feature trajectories. Specifically, the magnitude response is designed to modify the feature trajectory’s PSD function towards a reference PSD function. In this way, the temporal characteristics of the features are normalized. In the following texts, we will use the term PSD function to denote modulation spectrum.

The TSN framework consists of three steps as shown in Fig. 4. The first step involves the training of reference PSD functions from a group of clean utterances (see Fig. 4(a)). The reference PSD functions are used to represent the common temporal characteristics of the clean features. In the second step, temporal filters are designed for the utterance to be processed, one filter for each feature trajectory (see Fig. 4(b)). The final step is to filter the feature trajectories (see Fig. 4(c)). While the reference training step is performed offline, the filter design and filtering steps are performed on an utterance-by-utterance basis.

In both the training and processing steps, the $N$ feature trajectories are preprocessed individually by either the mean and variance normalization (MVN) or HEQ before the PSD estimation. The purpose of the preprocessing is to reduce the feature variations and therefore make the features more suitable for temporal filtering. Such preprocessing is common when temporal filters are used to improve the features’ robustness for speech recognition [29, 56]. In the next two sections, we will describe the filter design process and the training of the reference PSD functions.

B. Filter Design

The filter design process is summarized in Table I. Interested readers are referred to [43] for an example of TSN filter design, including the outputs of various steps and the filter responses at different SNR levels. The filter design process is divided into two parts: the first part estimates the desired magnitude response of the filter with optional modification (steps 1-3 in Table I) and the second part designs the time domain finite impulse response (FIR) filter that implements the desired magnitude response (steps 4-7 in Table I).

The same filter design process is performed individually for each feature trajectory. We will use one feature trajectory
as an example to explain the process. Let \( P_{\text{ref}}(k,j) \) and \( P_{\text{test}}(k,j) \) denote the reference PSD function and the test PSD function of the \( j^{\text{th}} \) feature trajectory, respectively, and \( k \) is the modulation frequency index. To simplify the notation, we will drop the feature index \( j \) in the following text. The reference \( P_{\text{ref}}(k) \) is trained from clean utterances during the training step (Fig. 4(a)) and the \( P_{\text{test}}(k) \) is estimated from the current feature trajectory to be processed. To normalize \( P_{\text{test}}(k) \) to \( P_{\text{ref}}(k) \), the filter’s magnitude response should be:

\[
|H(k)| = \sqrt{\frac{P_{\text{ref}}(k)}{P_{\text{test}}(k)}}
\]  

(3)

The FIR filter weights are then found from \( |H(k)| \) by using the Windowing method [57].

To retain the phases of the feature trajectories and prevent relative phase shift among the trajectories, linear phase FIR filters are used. The filter structure is non-causal and the filter weights are symmetrical w.r.t. the central tap. Using the same filter length for all feature trajectories, the resulting filters will have the same group delay for all feature trajectories.

During the filter design process, there are two considerations:

1) Normalizing the trend of the PSD function only: Suppose the filter with response \( |H(k)| \) can be implemented ideally, the filtered feature trajectory will have the following PSD function:

\[
P_{\text{norm}}(k) = |H(k)|^2 P_{\text{test}}(k) = P_{\text{ref}}(k)
\]  

(4)

For speech recognition task, the normalization, however, should not be carried out exactly. As the modulation spectrum carries information about the spoken content, the exact normalization of the modulation spectrum to the reference will render the feature not discriminative. We should only normalize the overall shape of the modulation spectrum which is related to the recording environment, while leaving the details of the spectrum intact. In the filter design process, several steps are designed to avoid extreme normalization. The simplest approach is to use a smooth \( P_{\text{ref}} \) and \( P_{\text{test}} \) in the filter design process. To obtain smooth PSD functions, the autoregressive (AR) model-based Yule-Walker method [47] is used for PSD estimation instead of the Fourier transform. The smoothness of the PSD function is controlled by the order of the AR model and a small order of 6 is found to be sufficient. Another step to avoid extreme normalization is to use a short filter. As shown in steps 5 and 6 of Table I, the filter’s weights are truncated and scaled with a Hanning window. Truncation and windowing have a smoothing effect on the realized filter’s magnitude response. Due to these operations, the realized filter’s magnitude response will be smoother than the ideal one in (3) and only the global trend of \( P_{\text{test}}(k) \) is normalized.

2) Combination with other temporal filters: Combination of temporal filters sometimes results in better performance if the filters are complementary to each other. For example, the cascade of RASTA and ARMA filters in [29] and the combination of TSN and ARMA filters in [43] both produce better results than any single filter alone for the Aurora-2 task. To take advantage of other temporal filters, we can simply multiply their magnitude responses with that of the TSN filter (step 3 of Table I):

\[
|H'(k)| = |H(k)||G(k)|
\]  

(5)

where \( |H'(k)| \) is the combined magnitude response, and \( |H(k)| \) and \( |G(k)| \) are the magnitude responses of the TSN filter and other filters, respectively. \( |G(k)| \) can be any magnitude response as far as it can be realized by an FIR filter with a short filter length.

### C. Training of Reference PSD Functions

Similar to the role of reference histogram in the HEQ techniques [16–19], the reference PSD functions act as the reference for the normalization of the PSD functions of all feature trajectories, including those of the clean utterances. Intuitively, to fulfill their role, the reference PSD functions should be able to represent the general temporal structure of clean feature trajectories. In the TSN framework, the general temporal structure is obtained by estimating the reference PSD functions from averaging the PSD functions of feature trajectories over a larger number of utterances (see Fig. 4a). The averaging process reduces other factors affecting the PSD functions, such as spoken content and speaker’s characteristics. For each feature trajectory, the reference PSD function captures the feature’s unique temporal characteristics.

The data used to train the reference PSD functions are usually the same as those used for the training of the acoustic model of the speech recognition system.

### D. Summary and Discussion

The TSN filter is designed to normalize the PSD functions of the feature trajectories. As the noisier feature trajectories usually have flatter PSD functions than the cleaner feature trajectories (see Fig. 3), the resulting filters on average tend to smooth the noisier features more aggressively than the clean features. Hence, the TSN filter implicitly adapts to different environmental conditions.

The TSN filter shares similar motivation as other environment-adaptive temporal filters, such as the temporal

---

**TABLE I**

**SUMMARY OF THE TSN FILTER DESIGN PROCEDURES**

For the \( j^{\text{th}} \) feature trajectory of current utterance,

1) Estimate the PSD of the feature trajectory: \( P_{\text{test}}(k,j) \).
2) Find the desired magnitude response of the filter using (3) \( |H(k,j)| = \sqrt{P_{\text{ref}}(k,j)/P_{\text{test}}(k,j)} \).
3) Optional modification of the filter response using (5).
4) Find the filter’s weights using the inverse discrete Fourier transform (IDFT): \( w(\tau,j) = \text{IDFT}[|H(k,j)|] \).
5) Extract the central taps of \( w(\tau,j) \) to form \( w'(\tau,j) \).
6) Apply Hanning window on \( w'(\tau,j) \) to reduce the truncation effect.
7) Normalize the sum of the weights \( w'(\tau,j) \) to one.
smoothing filter (TES) [56] and the linear equalization filter [58]. However, their design and realization are different in many ways. For example, the TES is designed from auto-correlation functions and realized by an IIR filter; the linear equalization filter is adapted to each noise type while the TSN filter is adapted to each utterance.

IV. SEGMENT-BASED IMPLEMENTATION

The TSN filter presented in Section III is implemented utterance-by-utterance, which leads to a large processing delay. When short processing delay is desired or environmental conditions change rapidly within one utterance, it is more desirable to process the features in a segment-by-segment fashion. To achieve this, we propose a segment-based implementation of the TSN filter (we call it TSNseg for short) and discuss it in this section.

A. Implementation Scheme

The way of dividing an utterance into segments is similar to the framing scheme used in the STFT (see Fig. 5). The whole utterance is divided into multiple overlapping and equal-sized segments with the last segment usually longer. The shift between two neighboring segments should be smaller than the segment length to ensure a gradual change of feature statistics between segments. For each segment, TSN filters are designed as if the segment is an utterance. The filter design process is the same as that of the utterance-based TSN described in section III. The TSN filters of a segment are used to process the frames near the center of the segment with exceptions for the first and last frames as shown in Fig. 5.

The clean utterances used for reference PSD functions training should also be processed segmentally. The segment-based MVN and HEQ are implemented using the same segmenting scheme as the TSNseg. The PSD functions are averaged over segments rather than utterances to generate the reference PSD functions. The segment length and shift used during the reference PSD function training match those used for the TSNseg.

B. Considerations

Two important parameters of the TSNseg are the segment length and the segment shift. A longer segment provides more frames for better estimation of the feature statistics such as the PSD function. However, a longer segment also leads to poorer adaptation to environmental changes such as a sudden increase/decrease in SNR level. The appropriate length of the segment is determined experimentally and may be environment/database dependent. The other parameter, the segment shift, controls the computational cost and the gradualness of the change of the feature statistics between neighboring segments. Ideally, the segment shift should be one frame as in several existing segment-based normalization methods [19]. However, with a shift of one frame, the TSN filters need to be re-designed for every frame and the computation becomes too expensive. In our experimental results, we find that the performance of the TSNseg is not very sensitive to the segment shift. We report the choice of these two parameters in the next section.

V. EXPERIMENTS

A. Experimental Setup

The TSN filter is evaluated on both the small vocabulary Aurora-2 task [50] and the large vocabulary Aurora-4 task [59]. The Aurora-2 and Aurora-4 data are generated by artificially adding recorded noises to clean speech. The noises are typical real life noises, such as subway train and babble noises. In Table II-III, the details of these tasks and the corresponding recognizer settings are listed. These recognizer settings comply with mainstream settings in the noise robust speech recognition community. For both tasks, the acoustic models are trained with clean utterances.

Mel-frequency cepstral coefficients (MFCC) are used as the features for speech recognition. The MFCC features are extracted using the standard WI007 feature extraction program [50] delivered with the Aurora-2 task. The Fourier transform frame size is 25ms, or 200 samples at 8000Hz sampling frequency. The frame shift is 10ms, hence the frame rate is 100Hz. In total, 39 features, including the 13 static cepstral features and their delta and acceleration features, are used as raw features. If not otherwise stated, the cepstral energy feature c0 is used instead of the log energy for its good performance in MVN preprocessing [29]. The 39 raw feature trajectories are processed by the TSN framework and then used for system training and testing.

For the TSN filter, the Yule-Walker method is used to estimate the PSD functions of feature trajectories. The order of the AR model for PSD estimation is set to 6 to obtain proper level of details. A filter length of 33 taps is used for the evaluation if not otherwise stated. We will discuss the effect of the filter length later.

B. Evaluation on the Aurora-2 Task

In the Aurora-2 task, the performance of the utterance-based TSN filter is compared with that of two other popular temporal filters, the RASTA filter [28] and the ARMA filter in the MVA processing [29]. The order of the ARMA filter is set to 3 and the pole value of the RASTA filter is 0.94 as suggested in [28]. Two techniques are used as the preprocessing units of the temporal filters: the MVN and the HEQ.

The performances of the temporal filters are shown in Table IV. The results are averaged over the ten test cases of
Table II

<table>
<thead>
<tr>
<th>Nature of the task</th>
<th>Connected English digital string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling freq.</td>
<td>8000Hz</td>
</tr>
<tr>
<td>Avg. utterance length</td>
<td>≈ 1.8 seconds</td>
</tr>
<tr>
<td>Training data</td>
<td>8440 clean utterances</td>
</tr>
<tr>
<td>Test data</td>
<td>10 test environments (cases)</td>
</tr>
<tr>
<td></td>
<td>7 SNR levels per environment</td>
</tr>
<tr>
<td></td>
<td>1003 test utterances per SNR level</td>
</tr>
<tr>
<td></td>
<td>70070 utterances, 230181 words in total</td>
</tr>
<tr>
<td></td>
<td>11-word test vocabulary</td>
</tr>
</tbody>
</table>

Table III

<table>
<thead>
<tr>
<th>Nature of the task</th>
<th>Large vocabulary continuous English speech</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling freq.</td>
<td>8000Hz</td>
</tr>
<tr>
<td>Avg. utterance length</td>
<td>≈ 7.6 seconds</td>
</tr>
<tr>
<td>Training data</td>
<td>7138 clean utterances</td>
</tr>
<tr>
<td>Test data</td>
<td>14 test environments (cases)</td>
</tr>
<tr>
<td></td>
<td>166 utterances per environment</td>
</tr>
<tr>
<td></td>
<td>2234 utterances, 38010 words in total</td>
</tr>
<tr>
<td></td>
<td>5000-word test vocabulary</td>
</tr>
<tr>
<td>Recognizer</td>
<td>HMM Toolkit</td>
</tr>
<tr>
<td>Acoustic model</td>
<td>3 states triphone model</td>
</tr>
<tr>
<td></td>
<td>3150 tied states, 4 Gaussians per state</td>
</tr>
<tr>
<td>Language Model</td>
<td>Bigram model</td>
</tr>
<tr>
<td>Test parameters</td>
<td>10 word insertion penalty</td>
</tr>
<tr>
<td></td>
<td>16 language model weight</td>
</tr>
<tr>
<td></td>
<td>250 pruning threshold</td>
</tr>
</tbody>
</table>

Aurora-2 and 5 SNR levels from 0 to 20dB. From the table, the performance of the TSN filter is similar to that of the ARMA filter and better than that of the RASTA filter.

While the ARMA filter and the TSN filter produce similar improvements on the Aurora-2 task (see Table IV), they achieve this differently. The TSN filter adapts to changing environments. On average, the TSN filter applies more aggressive smoothing on noisier features (see examples in [43]). The ARMA filter on the other hand smooths all the feature trajectories identically. We now investigate if the TSN filter complements the fixed ARMA filter to improve the performance.

The effect of the ARMA filter is integrated into the TSN filter by multiplying the magnitude response of the two filters as shown in (5). With the combined magnitude response, the TSN filter not only normalizes the feature’s temporal structure, but also applies additional smoothing to the features. The performance of the TSN filter integrated with the effect of the ARMA filter is shown in Table V. The ARMA filters with four different orders (1-4) are used. From the table, it is found that the integrated case TSN+ARMA is consistently better than both the ARMA filter and the TSN filter alone. This shows that the two filters are complementary for the Aurora-2 task.

To ensure that the results of these tests are statistically significant, we carry out hypothesis test as follows. The objective of the test is to decide whether two accuracies $p_1$ and $p_2$ are significantly different. We set the null hypothesis $H_0 : p_1 = p_2$ and the alternative hypothesis as $H_1 : p_1 \neq p_2$. If $H_0$ is accepted, the difference between $p_1$ and $p_2$ is not statistically significant and vice versa. The test statistic for the hypothesis test is shown as follows [60]:

$$z = \frac{\sqrt{N}(p_1 - p_2)}{\sqrt{p_1(1-p_1) + p_2(1-p_2)}}$$

where $N$ is the number of words in the test and $N = 230181$ for Aurora-2 task (see Table II). If the values of $p_1$ and $p_2$ are both around 85%, the difference between them must be larger than about 0.39% to reject $H_0$ with a confidence level of 99%, i.e. the difference of any two accuracies in Table IV-V must be larger than 0.39% to be statistically significant. From Table V, the results of TSN+ARMA are always significantly better than those of ARMA and TSN alone.

C. Evaluation on the Aurora-4 Task

The TSN filter and the RASTA and ARMA filters are also evaluated on the large vocabulary Aurora-4 task. Here, the best order of the ARMA filter is found to be one. The pole of the RASTA filter is 0.94.

The performance of the temporal filters is shown in Table VI. The results are averaged over the 14 test cases of the task. From the table, all the temporal filters improve the performance over the baseline systems. The TSN filter yields better results as compared to both the RASTA and the ARMA filters. It is also observed that the results with HEQ preprocessing are consistently better than the results with the MVN preprocessing on the Aurora-4 task, but not on the Aurora-2 task. This may be due to the fact that Aurora-4 has longer utterances so that the histograms of the features are better estimated for HEQ.

The statistical significance of the results in Table VI is evaluated. The accuracies in Table VI are in the range of 60%-70%, and the number of test words is 38010. By using the statistic in (6), the difference between two accuracies must be at least 0.96% to be significant with a confidence level of 95%. From Table V, the results of TSN are always significantly better than those of ARMA and RASTA.
TABLE VI
OVERALL RECOGNITION ACCURACIES (%) ON THE AURORA-4 TASK. THE PREPROCESSING IS FOLLOWED BY THE TEMPORAL FILTERS. NONE DENOTES NO TEMPORAL FILTERING.

<table>
<thead>
<tr>
<th>Preprocessing</th>
<th>Temporal Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td>MVN</td>
<td>60.75</td>
</tr>
<tr>
<td>HEQ</td>
<td>64.84</td>
</tr>
</tbody>
</table>

TABLE VII
RECOGNITION ACCURACIES (%) OF DIFFERENT FEATURES ON THE AURORA-2 TASK. MVN IS USED AS THE PREPROCESSING UNIT FOR ALL RESULTS.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Temporal Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td>MFCC+c0</td>
<td>78.49</td>
</tr>
<tr>
<td>MFCC+logE</td>
<td>77.06</td>
</tr>
<tr>
<td>PLP</td>
<td>78.44</td>
</tr>
</tbody>
</table>

Comparison of the Aurora-2 and Aurora-4 tasks gives us some insights into the filtering of speech features for robust speech recognition. While feature smoothing can reduce the feature mismatches, it also leads to removal of useful speech information. The degree of smoothing is a trade-off between these two factors and usually depends on the signal conditions such as the SNR level and the task complexity. Our experimental results show that the small vocabulary Aurora-2 task can be aggressively smoothed while the large vocabulary Aurora-4 task can only be mildly smoothed. This is why the ARMA filter uses a small order of one in the Aurora-4 task, i.e. less aggressive smoothing, while its order is three in the Aurora-2 task. Similarly, while the combination of the ARMA and TSN filters improves the recognition performance further by introducing additional smoothing to the TSN filter on the Aurora-2 task, such combination degrades the performance on the Aurora-4 task (Results showing degradation are not presented due to page limitations).

D. Evaluation on Different Features

The performance of the TSN filter may vary with different features. In this section, we evaluate the TSN filter on two other features, i.e. the MFCC features with log energy (MFCC+logE) and the PLP features [46] on the Aurora-2 task. The log energy is computed by the WI007 feature extraction program [50] and the PLP features are generated using the Matlab implementation of PLP [61]. For the PLP, the 13 PLP static cepstral features and their delta and acceleration are used. The preprocessing used is MVN.

The performance of the TSN filter for different features on the Aurora-2 task is shown in Table VII. For comparison, the results of the MFCC features with c0 (MFCC+c0) is also shown. The ARMA filter here uses an order of three and the RASTA filter’s pole is 0.94. From the table, the improvements of the temporal filters for the MFCC+logE and PLP show a similar range as those of the MFCC+c0. The TSN filter alone may not produce the best performance. However, the integration of the TSN with the ARMA filter (TSN+ARMA) always improves the performance.

E. Segment-based Implementation

The proposed segment-based implementation of the TSN filter (TSNseg) is evaluated on the Aurora-4 task as the utterances in Aurora-4 are long enough to test the segment-based implementation. For the TSNseg, the silence of the utterance should not be longer than the segment length, otherwise the preprocessing MVN/HEQ and the TSNseg will normalize the statistics of the silence to the reference statistics. Hence, voice activity detection (VAD) is used to find the speech segments and extra silence frames are removed in the beginning and ending of each utterance before training and testing. Only 25 silence frames, or 250ms, are kept. The VAD is performed by decoding the training and testing data using the decoder of the HMM Toolkit (HTK), i.e. the HVite tool [62].

The performance of the TSNseg with MVN preprocessing and various segment lengths is shown in Fig. 6. From experiments, we find that it is a good tradeoff to set the segment shift to be half of the segment length. For comparison, the utterance-based MVN preprocessing (MVN), the utterance-based TSN filter (TSN), and the segment-based MVN preprocessing (MVNseg) are also evaluated. All the configurations operate on the speech after VAD. From the figure, we have two observations. One is that the best segment length for the TSNseg is about 2.2s. The other is that although the TSNseg produces better results than the TSN, its improvement over the MVNseg preprocessing (about 2%) is similar to the improvement of the TSN over the MVN preprocessing (1.7%). This suggests that the gain of TSNseg over TSN is mainly due to its use of MVNseg preprocessing.

The main advantage of the TSNseg is the reduction of the processing delay. The average processing delay of the TSNseg is half of the segment length. If the segment length is chosen to be 2.2 seconds, the average delay will be 1.1 seconds, which is much lower than the utterance length of the Aurora-4 that can be more than 10 seconds. In addition, with the segment-based implementation, the TSN filter can adapt much faster to run-time changing environments, and is thus suitable for low-delay speech recognition deployments.
F. Effect of Filter Length

In the TSN filter design process, the weights used for filtering are only the central part of the available weights (step 5 of Table I). The performance of the TSN filter is affected by the number of weights used, or the filter length. Since the TSN filter is implemented by an FIR filter, the flexibility of the realized magnitude response of the TSN filter is proportional to the length of the filter. A long filter can model the desired magnitude response better, but also means higher computational cost and a longer transition period for filtering. With a long transition period, many frames in the beginning and ending of each utterance cannot be filtered and this causes inconsistency in the temporal characteristics within a feature trajectory. Due to these considerations, a short filter is preferred if it does not degrade the performance too much compared to a long filter.

In Fig. 7, the performance of the utterance-based TSN filter on the Aurora-2 task with MFCC features and different filter lengths is shown. Both the basic TSN filter (TSN alone) and the combination of the TSN filter with the ARMA filter (TSN+ARMA) are studied. From the figure, we found that although the optimal filter lengths for different configurations are different, the filter length of about 33 taps is a good choice. With 33 taps, or 330ms, the TSN filter can exploit temporal information on the scale of common syllables.

G. Contribution of Modulation Frequency Bands

It is generally believed that the intelligibility of human speech is closely related to the 1-16Hz modulation frequency range [36,37,39–42]. This suggests that the performance improvement of the TSN filter should mainly come from its gain in the 1-16Hz range. To verify this, we modify the magnitude response of the TSN filter to disable its gain in the high modulation frequency range. Specifically, we set the filter’s gain to one for the frequencies higher than a specified frequency (we call it edge frequency). Fig. 8 shows an example of modifying the TSN filter’s magnitude response. By studying the TSN’s performance with different edge frequencies, we are able to appreciate which part of the magnitude response contributes the most to the performance improvement.

In Fig. 9, the performance of the utterance-based TSN filter with different edge frequencies is shown. When the edge frequency is 50Hz, there is no modification to the gain of the TSN filter. From 50Hz downwards, the gain of the filter is set to one in more and more modulation frequencies and hence the performance of the filter decreases almost monotonically. When the edge frequency is 20Hz, the filter is disabled in the 20-50Hz range and the performance only drops about 1% from 84.4% to 83.4%. However, when the filter is also disabled in the 5-20Hz range, the accuracy drops significantly to below 79%. This observation suggests that the low modulation frequency range (<20Hz) of the magnitude response contributes much more to the improvement of the
TSN filter than the high modulation frequency range (>20Hz).

VI. Conclusions

This paper studies the temporal structure normalization (TSN) filter for speech recognition in noisy environments. The major contributions of the paper include: 1) it discusses the effects of additive noises on the speech modulation spectra with applications in feature extraction of speech recognition; 2) it conducts a comprehensive evaluation of the TSN framework with respect to different parameter settings; 3) it proposes a novel segment-based implementation of the TSN filter which significantly reduces the processing delay and increases the adaptability to changing environments. Generally, the TSN filter either performs comparably with or outperforms the state-of-the-art temporal filters on two popular benchmark tasks: Aurora-2 and Aurora-4.

The major advantage of the TSN filter is its ability to adapt to changing environments. As the magnitude response is designed to reflect the temporal characteristics of the feature trajectories that carry information about the signal distortion, it effectively reduces the feature-model mismatch during speech recognition.

The TSN filter is also a normalizing technique. From statistical point of view, the TSN normalizes the correlation of features over different frames, while other normalization techniques such as HEQ normalize the probability distributions of the features. Therefore, the TSN filter can be seen as presenting a new way to normalize the feature statistics to reduce the feature-model mismatch problem.

The TSN filter can be easily integrated into common speech recognition systems. The reference modulation spectra are estimated using the same training data that we use for acoustic modeling. The feature normalization takes place right after the feature extraction before acoustic training/decoding processes. It would be an interesting future work to study the interaction between feature normalization and acoustic modeling in speech recognition applications.

APPENDIX

A. Mathematical Formulation of Noisy Modulation Spectra

We now derive the modulation spectra of the noisy speech signals generated from the trajectories of filterbank coefficients. If we assume the noise is additive and statistically independent from the speech signal, the noisy speech signal can be represented as:

\[
y(i) = x(i) + n(i)
\]  

(7)

where \( y(i) \), \( x(i) \) and \( n(i) \) are the noisy speech, the clean speech and the noise, respectively; \( i \) is the speech sample index. Apply the STFT on both sides of (7), we get:

\[
Y(t, f) = X(t, f) + N(t, f)
\]  

(8)

where \( Y(t, f) \), \( X(t, f) \) and \( N(t, f) \) are the 2-dimensional short-time Fourier coefficients of the \( y(i) \), \( x(i) \) and \( n(i) \), respectively; \( t \) and \( f \) are the frame index and acoustic frequency index, respectively. The spectrogram of the noisy speech is

\[
|Y(t, f)|^2 = |X(t, f)|^2 + |N(t, f)|^2 + 2\Re\{X(t, f)(N(t, f))\}
\]  

(9)

where \( \theta(t, f) \) is the phase difference between \( X(t, f) \) and \( N(t, f) \) and represents the phase difference between the two. The phase item \( 2\Re\{X(t, f)(N(t, f))\} \) in (9) has zero expected value when noise and speech are independent, and are less significant than the other two items. Hence it is often omitted in many speech enhancement techniques such as spectral subtraction [4]. For simplicity, we ignore the phase item and (9) becomes

\[
|Y(t, f)|^2 = |X(t, f)|^2 + |N(t, f)|^2
\]  

(10)

The spectrogram is further processed by filterbank analysis, which integrates the frequency bins into filterbanks. Let \( F \) and \( F \) be the number of filterbanks and frequency bins respectively, and \( g_f \), \( f = 1, ..., F \) be the weights of the integration window for the \( l \)-th filterbank, with constraint \( \sum_{f=1}^{F} g_f = 1, l = 1, ..., L \). The calculation of the filterbank coefficients is as follows

\[
|Y^M(t, l)|^2 = \sum_{f=1}^{F} g_f |Y(t, f)|^2
\]  

(11)

where

\[
|X^M(t, l)|^2 = \sum_{f=1}^{F} g_f |X(t, f)|^2
\]  

(12)

\[
|N^M(t, l)|^2 = \sum_{f=1}^{F} g_f |N(t, f)|^2
\]  

(13)

and the superscript \( M \) denotes the filterbank domain.

From the filterbank trajectories, the modulation spectrum of the noisy speech is calculated by

\[
|Y_m(k, l)|^2 = |\text{DFT}(|Y^M(t, l)|^2)|^2
\]  

(14)

\[
= |\text{DFT}(|X^M(t, l)|^2 + |N^M(t, l)|^2)|^2
\]  

(15)

\[
= |X_m(k, l) + N_m(k, l)|^2
\]  

(16)

\[
= |X_m(k, l)|^2 + |N_m(k, l)|^2 + 2\Re\{X_m(k, l)(N_m(k, l))\} \cos(\beta(k, l))
\]  

(17)

where

\[
Y_m(k, l) = \text{DFT}(|Y^M(t, l)|^2)
\]  

(18)

\[
X_m(k, l) = \text{DFT}(|X^M(t, l)|^2)
\]  

(19)

\[
N_m(k, l) = \text{DFT}(|N^M(t, l)|^2)
\]  

(20)

\( k \) is the modulation frequency index and \( |Y_m(k, l)|^2 \), \( |X_m(k, l)|^2 \) and \( |N_m(k, l)|^2 \) are the modulation spectra of the noisy speech, the clean speech and the noise generated from the \( l \)-th filterbank, respectively. DFT represents the discrete Fourier transform, and \( \beta(k, l) \) is the angle between \( X_m(k, l) \) and \( N_m(k, l) \).
and $N_m(k,l)$. To make (14) easier to understand, we use a simpler representation:

$$P_y = P_x + P_n + Q$$

(18)

where $P_y$, $P_x$ and $P_n$ are the modulation spectra of the noisy speech, clean speech and noise respectively; $Q$ represents the phase item $2|X_m(k,l)|N_m(k,l)|\cos(\beta(k,l))$ in (14).

REFERENCES