

Deep neural network training for whispered speech recognition using small databases and generative model sampling

Shabnam Ghaffarzadegan¹ · Hynek Bořil^{1,2} · John H. L. Hansen¹

Received: 5 May 2017 / Accepted: 11 September 2017 / Published online: 26 October 2017
© The Author(s) 2017. This article is an open access publication

Abstract State-of-the-art speech recognition solutions currently employ hidden Markov models (HMMs) to capture the time variability in a speech signal and deep neural networks (DNNs) to model the HMM state distributions. It has been shown that DNN–HMM hybrid systems outperform traditional HMM and Gaussian mixture model (GMM) hybrids in many applications. This improvement is mainly attributed to the ability of DNNs to model more complex data structures. However, having sufficient data samples is one key point in training a high accuracy DNN as a discriminative model. This barrier makes DNNs unsuitable for many applications with limited amounts of data. In this study, we introduce a method to produce an excessive amount of pseudo-samples that requires availability of only a small amount of transcribed data from the target domain. In this method, a universal background model (UBM) is trained to capture a parametric estimate of the data distributions. Next, random sampling is used to generate a large amount of pseudo-samples from the UBM. *Frame-Shuffling* is then applied to smooth the temporal cepstral trajectories in the generated pseudo-sample sequences to better resemble the temporal characteristics of a natural speech signal. Finally, the pseudo-sample sequences are combined with

the original training data to train the DNN–HMM acoustic model of a speech recognizer. The proposed method is evaluated on small-sized sets of neutral and whisper data drawn from the UT-Vocal Effort II corpus. It is shown that phoneme error rates (PERs) of a DNN–HMM based speech recognizer are considerably reduced when incorporating the generated pseudo-samples in the training process, with + 79.0 and + 45.6% relative PER improvements for neutral–neutral training/test and whisper–whisper training/test scenarios, respectively.

Keywords Speech recognition · Deep neural networks · Gaussian mixture models · Random sampling · Small datasets

1 Introduction

Deep neural networks (DNNs) have been successfully deployed in automatic speech recognition (ASR) primarily due to the recent advances in machine learning methods and computer hardware. It has been shown by many researchers that a hidden Markov model (HMM) with DNN models (DNN–HMM) has the potential to model sequences of acoustic observations better than an HMM system with Gaussian mixture model (GMM) states (GMM–HMM) in a variety of tasks, including very large databases and large vocabularies (Dahl et al. 2012; Deng et al. 2013; Hinton et al. 2012; Seide et al. 2011). While traditional GMMs have been a convenient means of modeling HMM state distributions for decades (Liporace 2006; Rabiner (1990; Young 1996), DNNs have recently demonstrated a superior quality in modeling data space nonlinearities (Hinton et al. 2012). In speech systems, DNN models employ multiple hidden layers and a large output layer mapped to senone states. While

✉ John H. L. Hansen
john.hansen@utdallas.edu

Shabnam Ghaffarzadegan
shabnam.ghaffarzadegan@utdallas.edu

Hynek Bořil
borilh@uwplatt.edu

¹ Center for Robust Speech Systems (CRSS), University of Texas at Dallas, Richardson, TX 75083-0688, USA

² Electrical Engineering Department, UW-Platteville, Platteville, WI 53818-3099, USA

the DNN–HMM paradigm becomes increasingly popular for acoustic modeling in ASR engines as well as other speech-oriented applications, there are still numerous instances where a GMM–HMM is arguably a better choice. Due to their parametric nature, GMMs tend to be far more conservative in terms of the amount of training data required to learn and generalize the underlying structure of the modeled classes, and moreover, are easily adaptable to new scenarios when seeing small amounts of target-domain samples. On the contrary, DNNs tend to underperform or completely fail in situations where only limited sample sizes are available for training (Lasserre et al. 2006; Ng and Jordan 2002).

For a DNN model, having access to an excessive amount of training samples is a major factor in order to capture the non-linearity in the data structure and allow the model/training algorithm to converge. However, there are numerous speech applications where only small training or adaptation sets are available and acquiring more data is either challenging or not feasible, such as in specialized medical recordings [e.g., bladder cancer (Li et al. 2007b)], or alternative speech modes for communication such as whisper, where acquiring and transcribing larger amounts of data is costly and tedious (Ghaffarzadegan et al. 2014a, b, 2015, 2016).

To address insufficient data sizes in training, one solution could be adding artificial samples to the training set. Recently, there have been several studies on using small datasets for training neural networks (NNs). In Mao et al. (2006), a NN training method based on posterior probabilities was proposed. In Li et al. (2007a), a mega-diffusion function was introduced to train the NN for dynamic manufacturing environments. The authors in Huang and Moraga (2004) proposed an information diffusion technique to derive new samples from a fuzzy set. However, none of these studies have considered DNNs operating in such a complex optimization space as seen in speech recognition systems. Production of artificial training samples in the context of speech recognition was considered in Bou-Ghazale and Hansen (1994), (1996), and shown to provide gains when only limited stressed speech samples were available for training, but the approaches relied on excessive a priori knowledge of the underrepresented target stressed speech domain and focused solely on accommodating GMM–HMM models. The goal of our study is to bridge the gap between DNN–HMM and GMM–HMM in the cases where the available data is sufficient for successful GMM training but at the same time too limited to extract meaningful DNN models.

Studies on generative and discriminative classifiers (Ng and Jordan 2002; Rubinstein and Hastie 1997) suggest that generative models tend to be more successful in learning from small datasets as far as balanced/representative samples of the underlying data structure are available, and when the modeled process matches the assumptions made by the generative model (e.g., assumptions about the structure of

the modeled distributions). Inspired by this, we introduce a new pseudo-sample generation scheme that utilizes a generative GMM trained on a small training dataset to produce large amounts of artificial samples. The artificial samples are then supplied to a DNN to learn the generalized structural information captured by the GMM.

The method proposed in this study uses only a small amount of data from the target domain to produce large quantities of pseudo-samples. The method utilizes a simple sampling algorithm to generate the pseudo-samples from the generative GMM model trained on the target data. The generated pseudo-samples are combined with the original target domain data and used to train a DNN–HMM recognizer. It will be shown that supplying the DNN–HMM system with the artificial samples considerably reduces recognition errors when compared to a DNN–HMM trained only on the original training data, and moreover, yields an acoustic model that surpasses a GMM–HMM trained on the original samples.

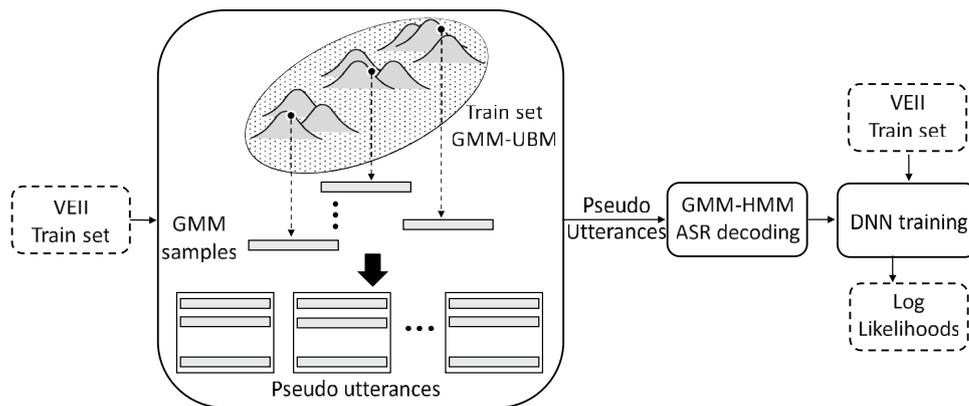
This paper is organized as follows. First, we introduce the method for producing artificial samples for DNN training. Next, the speech corpora used in the evaluations are briefly described. Finally, the effectiveness of the proposed method is evaluated on neutral and whisper datasets of the UT-Vocal Effort II corpus.

2 Proposed method

A number of speech recognition studies have demonstrated the effectiveness of DNNs in modeling state emission distributions in HMMs (Dahl et al. 2012; Hinton et al. 2012; Seltzer et al. 2013). However, DNN modeling accuracy reduces dramatically when faced with insufficient training data. A straightforward solution is to provide additional labeled data that would contain the task-relevant information in a quantity that makes the information accessible to the DNN model through the learning process. Encouraged by the DNN success on large databases, in this section we propose a strategy to produce a large population of pseudo-samples from small available datasets. The goal of the method is to improve DNN–HMM learning from limited data and enable effective use of DNN–HMM acoustic models in constrained resource scenarios that have been so far widely dominated by GMM–HMM systems. Figure 1 outlines the proposed approach.

In the initialization step, a universal background model (UBM) is trained on the available small training dataset. The UBM is a GMM that captures general speech characteristics of the training samples in terms of a multimodal statistical distribution. In this study, a GMM with diagonal covariance matrices is used to model the distribution of Mel frequency cepstral coefficients (MFCC Davis and Mermelstein 1980)

Fig. 1 DNN training using pseudo-samples generated from GMM-UBM



extracted from the speech signal. A diagonal rather than a full covariance matrix is used, reflecting the common choice in acoustic modeling for ASR, and following the usual assumption that MFCC feature vectors are reasonably decorrelated. Subsequently, pseudo-samples that reflect the training set distribution can be drawn from the GMM.

2.1 Random sampling of a GMM-UBM

There are various sampling methods available in the literature for synthesizing observations from sample distributions. A simple random sampling from the cumulative distribution function (CDF) is the most straightforward way of drawing samples. However, this method is directly applicable only when a closed form of the probability density function and its integral are known. However, in many real-world problems, the closed form representations are not available and other sampling methods need to be used. As an alternative, the Markov chain Monte Carlo (MCMC) algorithm is one of the most popular approaches for sampling multivariate probability density functions. Here, a Markov chain and random walks are utilized to numerically sample the CDF (Brooks 1998). Various modifications of the MCMC method are also available, such as the Gibbs sampling (Casella and George 1992) that generates samples from the marginal distribution, or the hybrid Monte Carlo (HMC) method that eliminates the need for random walks during the state space search (Neal 2010).

In our study, we devise a simple sampling algorithm that leverages the fact that the target posterior distribution is modeled by a GMM. In the GMM-UBM, the probability density function for the feature vector at time t , $\mathbf{y}(t)$, is modeled by a mixture of weighted Gaussians:

$$f(\mathbf{y}(t)) = \sum_{m=1}^M w_m N(\mathbf{y}(t); \mu_m, \Sigma_m), \tag{1}$$

where $N()$ represents the normal distribution function; M is the number of Gaussian mixture components; μ_m and Σ_m are

the m -th mixture component vector of means and covariance matrix, respectively, and w_m is the component weight.

To draw a sample from this distribution, we use the algorithm detailed in Table 1, which consists of two stages: (i) random Gaussian component selection, and (ii) within-component sampling. The task of the first stage (A) is to randomly select one component from the pool of the GMM Gaussians. While the selection is random, it has to reflect the distribution of component weights; components with higher weights will be chosen more frequently than those with lower weights. More specifically, with increasing number of draws, the normalized frequency with which a certain component is selected over others needs to approach the component weight (note that component weights within a GMM sum up to 1). To assure this, in Table 1, Step A.1, we define a so-called *Cumulative Weight Function* (CWF) which integrates component weights. To randomly select a component, we draw a random number between 0 and 1 from a uniform distribution (Step A.2), and use it as an argument to the inverse CWF (Step A.3). This sampling procedure is directly inspired by the traditional CDF sampling, where in our case the CDF is replaced by the CWF and the output k of the inverse CWF represents the index of the selected mixture component rather than the final sample.

In the second stage (B), a random sample from the selected k -th component is drawn. First (Step B.1), a random sample is drawn from a multivariate standard normal distribution, i.e., a normal distribution with zero means and an identity covariance matrix. Subsequently (Step B.2), the sample is scaled, dimension-wise, by the standard deviations captured in the square-rooted diagonal covariance matrix, and superposed with the vector of means of the k -th mixture component.

With repeated draws, the sampling procedure will be producing observations whose distribution will approach the one captured in the GMM-UBM. Since the UBM captures an acoustic-phonetic space of the training speech samples, phones present more prominently in the training set will

Table 1 Random sampling from a Gaussian mixture model

A) Random Gaussian component selection:

1) Compute a *Cumulative Weight Function* defined as:

$$CWF(m) = \sum_{i=1}^m w_i;$$

2) Draw a random sample x from a uniform distribution:

$$U(0, 1) \mapsto x;$$

3) Select a corresponding Gaussian mixture component k using the inverse of *CWF* defined as:

$$k = CWF^{-1}(x) = \arg \min_m |CWF(m) - x| \quad m = 1, \dots, M$$

B) Within-Component Sampling:

1) Draw a random sample \mathbf{r} from a standard multivariate normal distribution

$$N(\boldsymbol{\mu}_0, \mathbf{I}_n) \mapsto \mathbf{r},$$

where $\boldsymbol{\mu}_0$ is a vector of zero means, $\boldsymbol{\mu}_0 = (0, 0, \dots, 0)$,

and \mathbf{I}_n is an identity matrix $\mathbf{I}_n = \text{diag}(1, 1, \dots, 1)$;

2) Extract a pseudo-sample from the k -th component of the Gaussian mixture model:

$$\mathbf{y}_{\text{Pseudo}} = \mathbf{r} \times \sqrt{\text{diag}(\boldsymbol{\Sigma}_{k,0}, \boldsymbol{\Sigma}_{k,1}, \dots, \boldsymbol{\Sigma}_{k,D-1})} + \boldsymbol{\mu}_k$$

where $\boldsymbol{\mu}_k$ is the vector of means and $\boldsymbol{\Sigma}_{k,0 \dots D-1}$ the D

diagonal entries in the covariance matrix of the k -th mixture component, and \times is the matrix product.

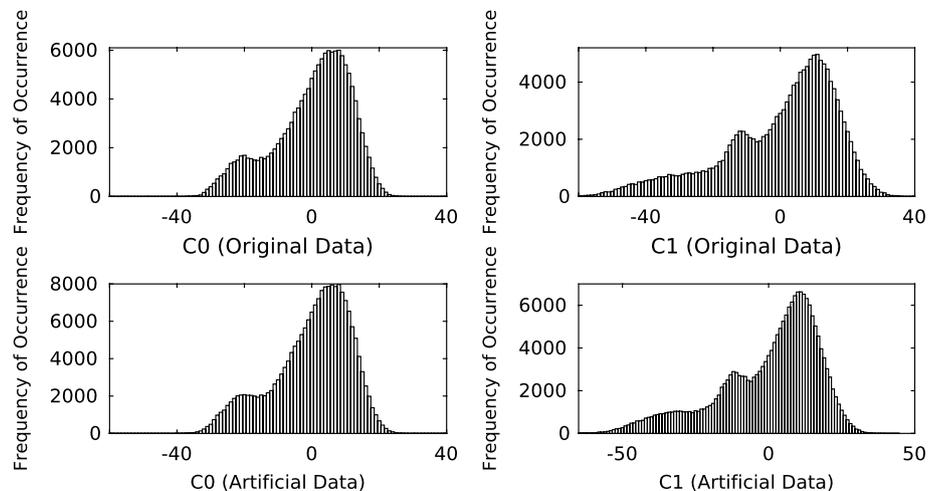
have a greater impact on the overall UBM statistics. In turn, random sampling from the UBM will produce samples that will reflect the phonetic balance seen in the original training set.

An example of the training dataset histograms and histograms of the pseudo-samples produced using the proposed sampling procedure is shown in Fig. 2. Here, the first two cepstral coefficients c_0 and c_1 extracted from a UT-Vocal Effort II training set (introduced in Sect. 3.1) are presented in the upper plots and the histograms of the generated pseudo-samples in the bottom plots. It can be seen that the

histogram contours for the generated data are approaching the ones of the original real speech.

After generating the artificial speech feature vectors, the sampled data are grouped into sequences to form pseudo-utterances. The pseudo-utterances are then fed to a GMM–HMM ASR engine trained on the original labeled training set to estimate state label alignments and provide target state emission probabilities for the DNN training. This is a common step required for DNN–HMM training. The slight difference is that the orthographically labeled training data used in typical ASR engines need only a forced alignment pass, while in our case the orthographic, and hence also phonetic, transcriptions of the

Fig. 2 Cepstral distributions in neutral training set and in neutral pseudo-utterances sampled from GMM–UBM



pseudo-utterances are not a priori known and need to be estimated through a decoding pass.

2.2 Smoothing pseudo-utterances

The generated pseudo-utterances contain random sequences of samples drawn from the GMM–UBM. While the individual samples are produced to resemble real speech feature vectors, their sequential order is random. In the real speech signal, each phone segment is typically represented by at least several consecutive feature frames. The random sampling procedure does not impose any constraints on inter-frame phone transitions. As a result, the pseudo-utterances are likely to contain abrupt phone transitions between adjacent frames. At the same time, the individual samples in the pseudo-utterances already contain meaningful first- and second-order time derivatives, as those are modeled together with the static coefficients by the GMM–UBM. In this way, the state models trained on the pseudo-utterances will be still be exposed to a source of realistic temporal information. Since the pseudo-utterances are to be decoded by a GMM–HMM recognizer in order to estimate frame-level phone labels and state emission probabilities, it seems meaningful to first adjust the temporal cepstral trajectories in the pseudo-utterances to better resemble those in real speech accommodate the GMM–HMM transition models. In addition, it may be also helpful to adjust the ratio of the acoustic model versus phone-level language model (LM) weights in the pseudo-utterance decoding in favor of the acoustic weight as the LM will tend to favor hypotheses with phone transitions seen in real speech over the actual rare transitions captured in the pseudo-utterances.

In the following section, two approaches to temporal smoothing of cepstral trajectories are discussed.

2.2.1 RASTALP filtering

Relative spectral filtering (RASTA) (Hermansky and Morgan 1994) uses a band-pass filter to eliminate the components in the speech signal that vary either too slow (e.g., the DC component) or too fast to be related to the speech content. RASTA is typically applied either on the log energy outputs of the feature extraction filterbank or directly on the cepstral vectors, which is the choice in our study. Assuming filtering in the cepstral domain, the high-pass portion of the RASTA filter eliminates the DC bias and the low-pass smoothens the cepstral trajectories and suppresses abrupt amplitude changes. Recently, a modified RASTA filter, RASTALP (Bořil and Hansen 2011), has been introduced and shown to outperform the original RASTA in ASR under various adverse conditions (Bořil et al. 2011). RASTALP reduces the original high-order band-pass filter to a low-order low-pass filter, effectively alleviating

transient distortions of the original while enabling engagement of other normalizations (e.g., cepstral mean normalization (CMN) (Atal 1974) to eliminate the DC component). The transfer function of the second-order infinite-impulse response (IIR) RASTALP filter is defined:

$$H(z) = \frac{\sum_{m=0}^M b_m}{\sum_{n=0}^N a_n} \quad (2)$$

where the filter coefficients are $\mathbf{B} = [b_0, b_1, b_2] = [0.10408, 0.20816, 0.10408]$ and $\mathbf{A} = [a_0, a_1, a_2] = [1, -0.90342, 0.31973]$, assuming a 10 ms frame step (Bořil and Hansen 2011). RASTALP filtering has a good potential to smoothen abrupt frame-to-frame trajectory changes in pseudo-utterances, making them closer to real speech. At the same time, linear filtering will impact the pseudo-sample distributions and make them depart from the GMM–UBM (note that this will be the case even when the original speech samples are also RASTALP-filtered prior to building the GMM–UBM). In summary, RASTALP filtering may have both positive and negative contribution to the effort of making pseudo-utterances more realistic. Sect. 3.5 will experimentally investigate which of those will prevail in terms of ASR performance.

2.2.2 Frame-shuffling

As discussed in the previous section, linear filtering of temporal trajectories in pseudo-utterances may yield smoother and more natural cepstral contours that will better fit transition models in a GMM–HMM trained on real speech. Unfortunately, the filtering will also impact the distributions of the processed samples and may increase their mismatch with the GMM–HMM state models. In this section, we introduce an algorithm called *Frame-Shuffling* which aims at making the temporal contours in pseudo-utterances more realistic while keeping the cepstral distributions intact. As a result, the temporal trajectories may better fit the transition models without the decoding accuracy of the state models being jeopardized.

Frame-Shuffling reorders the sequence of pseudo-utterance frames so that the distribution of their between-frame Euclidean distances would approach the one seen in real utterances. Clearly, reordering frames will have no impact on the overall frame distribution. The *Frame-Shuffling* algorithm is detailed in Table 2. To guide the shuffling process, a single Gaussian model is first estimated for Euclidean distances between neighboring frames in real training utterances (Steps 1, 2). Note that in Step 1, I denotes, for simplicity, the total number of feature frames in the training set. In reality, the number of measured distances would be slightly reduced as between-utterance distances (i.e., distances

Table 2 The Frame-Shuffling algorithm

Shuffling pseudo-utterance frames:

- 1) Extract euclidian distance for adjacent frame-level feature vectors for all real utterances:

$$D_i = d(\mathbf{y}_i, \mathbf{y}_{i+1}) \quad i = 0, 1, \dots, I - 2;$$

- 2) Estimate a Gaussian pdf for D :

$$f_D = N(D; \mu_D, \sigma_D^2)$$

- 3) Input: unshuffled sequence of pseudo-utterance frame vectors

$$\mathbf{Y}_{Pseudo} = (\mathbf{y}_{Pseudo,0}, \mathbf{y}_{Pseudo,1}, \dots, \mathbf{y}_{Pseudo,L-1});$$

- 4) Shuffling initialization: $\mathbf{y}_{Shuffled_Pseudo,0} = \mathbf{y}_{Pseudo,0}$;

- 5) For $l = 0 : L - 2$,

- (i) Draw a random distance sample from

$$N(\mu_D, \sigma_D^2) \mapsto D_{Pseudo,l}$$

- (ii) Find a frame vector from \mathbf{Y}_{Pseudo} that has not yet been included in $\mathbf{Y}_{Shuffled_Pseudo}$ and that has the closest distance to $D_{Pseudo,l}$ from the current $\mathbf{y}_{Shuffled_Pseudo,l}$:

$$q(l) = \arg \min_q |d(\mathbf{y}_{Shuffled_Pseudo,l}, \mathbf{y}_{Pseudo,q}) - D_{Pseudo,l}|$$

$$q = 0, \dots, L - 1 \wedge q \neq q(k), \quad k = 0, \dots, \max(l - 1, 0);$$

- (iii) Add the frame vector in the $\mathbf{Y}_{Shuffled_Pseudo}$ sequence:

$$\mathbf{y}_{Sorted_Pseudo,l+1} = \mathbf{y}_{q(l)}$$

- (iv) Next;

- 6) $\mathbf{Y}_{Shuffled_Pseudo}$ has been completed.

between the last frame of one utterance and the first frame of a subsequent utterance) are excluded.

The shuffling procedure is initiated by taking the first sample from the unshuffled frame sequence and using it as a starting anchor point in the shuffled sequence (Steps 3, 4). Subsequently, a random distance $D_{Pseudo,l}$ is drawn from the distance probability density function (PDF) f_D (Step 5.i) and a sample from the unshuffled sequence whose distance from the current anchor vector is closest to $D_{Pseudo,l}$ is found and added to the shuffled sequence (Steps 5.ii and 5.iii), becoming the new anchor point for the next distance matching. Throughout the shuffling algorithm, samples that were already included in the shuffled sequence are automatically excluded from the subsequent searches (see the search boundaries in Step 5.ii). The procedure is repeated from Step 5.i until the completion of the shuffled sequence (Step 6).

While the *Frame-Shuffling* algorithm is intended to be intuitive and straightforward, the authors experienced several issues during its implementation and testing. The measured distance histograms for the original data had ‘well behaved’ unimodal contours that resemble a slightly skewed normal distribution (or more precisely, a generalized extreme value distribution – GEV), suggesting that a single Gaussian PDF would provide for a reasonable distance model. After *Frame-Shuffling*, the distance histograms of the pseudo-utterances would in most part match the shape of the target Gaussian distribution f_D , however, there was always an additional left side lobe emerging close to the y-axis. This is a direct consequence of drawing distance samples from a PDF whose

left tail extends to negative values (which is unavoidable for Gaussian PDFs). This causes *Frame-Shuffling* to occasionally draw negative $D_{Pseudo,l}$. Given that samples with negative Euclidean distances cannot exist and the nearest meaningful (i.e., nonnegative) Euclidean distance is zero, the algorithm will automatically reach for a sample with a minimum distance from the current anchor. This will be the case for all negative $D_{Pseudo,l}$, causing occasional plateaus in the temporal trajectories and building up an additional side lobe in the distance histogram. To address this, we have added a constraint on the draw that any $D_{Pseudo,l} < Th$ would be discarded and a new draw would follow. The threshold value was experimentally set to $Th = 8$, following the minimum distance values seen in the real speech GEV-like histogram.

In addition, we have noticed that once *Frame-Shuffling* is getting close to its completion, the fewer samples left to be shuffled, the greater departure of the available distances from $D_{Pseudo,l}$ can be expected. In particular, we have observed that samples gathered in a tight Euclidean space tend to pile up towards the end of the shuffling process, contributing once again to the unwanted side lobe in the vicinity of the y-axis, in spite of the Th constraint introduced in the previous paragraph. To alleviate this effect and impose a greater variability in the leftover vectors, we have loosened the search criterion in Table 2, Step 5.ii; instead of searching for the best match with $D_{Pseudo,l}$, the first sample in the sequential search whose distance is within $\pm 5\%$ of $D_{Pseudo,l}$ is taken. While this will result in a slight departure of the shuffled PDF from f_D , it will limit

the production of outlier distances towards the end of the shuffled sequences, keeping the trajectory dynamics in a range typical for real speech (see examples in Sect. 3.5).

It is noted that *Frame-Shuffling* is a simple dynamic programming algorithm that, due to its sequential search nature, provides suboptimal ordering of the feature frames with respect to the desired global between-frame distance distribution. More sophisticated methods that would minimize the global distance error costs over the whole pseudo-utterance could further improve the shuffling procedure. However, as shown in Sect. 3.5, the current algorithm is quite effective in approximating the desired distance characteristics while requiring only a minimal computational overhead.

After the temporal smoothing is completed using either RASTALP or *Frame-Shuffling*, state labels and emission probabilities are extracted via GMM–HMM decoding and used for DNN training.

3 Experimental results

The following sections experimentally evaluate the proposed sampling and smoothing algorithms in terms of ASR performance. Sample distributions and temporal trajectories in the real speech samples and produced pseudo-utterances are also studied.

3.1 UT-vocal effort II corpus

The UT-Vocal Effort II (VEII) corpus (Zhang and Hansen 2009) is used for evaluations in this study. The corpus captures neutral (modal) and whispered speech recordings, which allows us to test the proposed algorithms in two substantially different scenarios. Whisper is an example of a strongly underrepresented speech modality in terms of publicly available resources, and hence may provide a practical insight into the applicability of the suggested schemes.

The corpus captures *read* and *spontaneous* speech from 112 speakers (37 males and 75 females). The *spontaneous* part of the corpus was recorded in a simulated cyber cafe scenario inside a 12' × 12' sound booth where two people were communicating in neutral and whispered modes and a third person was trying to overhear as much key information as possible. The *read* portion of the corpus contains three parts: (i) whole sentences – 41 TIMIT sentences (Zue et al. 1990) read in neutral and whispered modes; (ii) whispered words – two paragraphs from a local newspaper read in modal voice, with some *words* being whispered; (iii) whispered phrases – two paragraphs from a local newspaper read in modal voice, with some *phrases* being whispered. In our experiments, a subset of the read part of the VEEI corpus

– neutral and whispered whole sentences from 39 females and 19 males are utilized. All recordings were downsampled from 44.1 to 16 kHz/16 bits.

3.2 Experimental setup

In all experiments, gender-independent ASR models are built using the state-of-the-art ASR toolkit the Kaldi (Povey et al. 2011). In the GMM–HMM setup, three-state left-to-right triphone HMMs with GMM are used to model 48 English phones including silence. The number of mixture components per state is optimized by the Kaldi's training algorithm and varies across states. 13 static MFCCs and their first- and second-order time derivatives are extracted using a 25 ms window shifted with a skip rate of 10 ms. All feature vectors are mean-normalized (Atal 1974).

GMM–HMM training is divided into two stages, a *preliminary stage* and *the main stage*. In the *preliminary stage*, a basic triphone GMM–HMM recognizer is trained on the training dataset and subsequently used to label the training data into classes representing clustered HMM states. The state-labeled training data are then used to train a linear discriminant analysis (LDA) (Haeb-Umbach and Ney 1992) which takes sequences of 9 consecutive feature vectors as an input. Once the LDA is established, it is applied on all training feature vectors. This whole procedure is repeated one more time to train the maximum likelihood linear transform (MLLT) (Gales 1998). Here, the LDA-transformed features are taken as an input. Similarly as in the LDA training, state alignments from a new GMM–HMM recognizer trained on the LDA-transformed features are used as targets for the MLLT training.

In the *main stage*, the LDA and MLLT transforms are hardwired in the feature extraction front-end for feature decorrelation and dimensionality reduction. The LDA–MLLT processed training samples are used to train the resulting GMM–HMM system using speaker adaptive training (SAT) (Matsoukas et al. 1997).

In the DNN–HMM system, a neural network with three hidden layers is used. Each hidden layer contains 2048 nodes with p-norm ($p = 2$) activation functions (Zhang et al. 2014). The layers are pretrained in a generative fashion using layer-wise supervised backpropagation (Zhang et al. 2014). The output layer comprises 5175 neurons with softmax nonlinearities. The DNN takes 9-frame segments of the MFCC vectors transformed by LDA–MLLT as its input. When using the original real speech training data, the target HMM state probabilities for DNN training are extracted via *forced alignment* with the GMM–HMM system. For pseudo-utterances, the target labels are obtained through a *decoding pass* with the GMM–HMM. Once the training is completed, the posterior probabilities from the DNN output

Table 3 Speech corpora statistics

		# Sessions					
Corpus	Set	Style	M	F	# Sents	Dur	# Wrds
VEII	Train	Ne	13	26	766	30	154
		Wh			779	45	159
	Test	Ne	5	13	351	14	152
		Wh			360	20	137

M/F males/females, *Train* training set, *Test* test set, *Ne/Wh* neutral/whispered speech, *#Sents* number of sentences, *Dur* total duration in minutes, *Open Speakers* different speakers in Train/Test

Table 4 Performance of GMM–HMM and DNN–HMM recognizers on neutral and whisper tasks; PER (%)

Train	Test	GMM–HMM	DNN–HMM	DNN–HMM Pseudo-samples
Ne-VEII	Ne-VEII	27.5	62.0	13.1
Wh-VEII	Wh-VEII	45.0	70.0	42.4

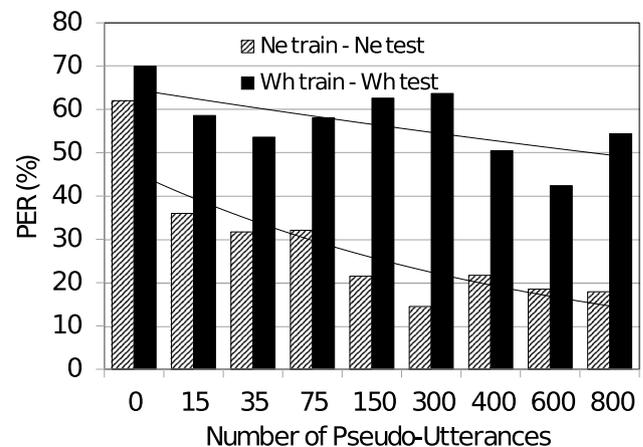
are converted to log-likelihoods to model the HMM state emission probabilities.

Both the GMM–HMM and the DNN–HMM training closely follow Kaldi's *TIMIT/s5* recipe.

3.3 Baseline experiments

In the initial experiment, a GMM–HMM recognizer is evaluated on the VEII neutral and whisper tasks. In the *neutral* task, the recognizer is trained and tested on the *Ne* VEII sets and in the *whisper* task on the *Wh* VEII sets (Table 3). As shown in the third column of Table 4, the recognizer reaches a phoneme error rate (PER) of 27.5% on the neutral task and 45.0% on the whisper task. The results are in agreement with the results in Ghaffarzadegan et al. (2014b), where whisper-trained recognizer tested on whisper data yielded a lower recognition rate compared to a neutral-trained model tested on neutral data. As discussed in Ghaffarzadegan et al. (2014b), the main cause of this disparity may be attributed to the higher confusability of the whisper phone set where voiced and unvoiced phone groups seen in neutral speech are now all mapped to the unvoiced acoustic space, causing wide overlaps of originally voiced and unvoiced fricative and stop pairs.

In the next experiment, performance of a DNN–HMM system is evaluated on the same neutral and whisper VEII tasks from the previous paragraph. It is noted, and this becomes especially prominent in the DNN case, that the available training data is very limited –30 min of neutral speech and 45 minutes of whispered speech (Table 3). As expected, these amounts are insufficient for the DNN

**Fig. 3** DNN–HMM performance as a function of the number of pseudo-utterances included in the training set; PER (%)

models to fully learn the acoustic-phonetic characteristics required for a successful speech recognition. As shown in the fourth column of Table 4, the DNN–HMM PERs reach 62.0% on the neutral task and 70.0% on the whisper task, notably lagging behind GMM–HMM.

3.4 DNN training with pseudo-samples

In this section, we study the effect of including pseudo-utterances in the DNN–HMM training set on ASR performance. The pseudo-utterances considered here are produced using the random sampling algorithm from Sect. 2.1 and exclude smoothing, which will be investigated separately in Sect. 3.5. In all experiments, a task-specific GMM–UBM with diagonal covariance matrices is trained on the respective training dataset (either *Ne* or *Wh*). Two aspects are studied: (i) DNN–HMM performance as a function of the *number of pseudo-utterances* added to the training set, and (ii) DNN–HMM performance as a function of the *number of mixture components* in the GMM–UBM used for the pseudo-utterance generation.

In the first experiment, we create training sets of various sizes by pooling together the original training set (either *Ne* or *Wh*) and a varying number of pseudo-utterances with

300-frame length. The GMM–UBM used here comprises 30 Gaussian mixture components and each generated pseudo-utterance has a fixed length of 400 frames (4 seconds). Figure 3 shows performance of the DNN–HMM systems as a function of the number of pseudo-utterances included in the training set. The leftmost result, ‘0’ pseudo-utterances, refers to the baseline scenario from Sect. 3.3 where only the original training data is used. It can be seen that by adding as few as 15 pseudo-utterances, the Ne PER dropped from 62.0 to 36.0% and the Wh PER from 70.0% to 58.6%.

The results also show that incrementing the number of pseudo-utterances in the training set does not have a monotonic impact on the DNN–HMM accuracy, as there are multiple local minima and maxima in both the neutral and whisper PER trends. This is likely caused by the inherent randomness in the sample generation and the related fluctuating quality of the pseudo-utterances in terms of their resemblance of real utterances, especially with respect to their temporal characteristics, as well as the convergence of the DNN training process which is bound to settle in local optima. This being said, Fig. 3 shows descending global trends for both the Ne and Wh PERs.

In the neutral task, the performance is improving rapidly with the number of added pseudo-samples and reaches minimum at 14.5 % PER for 300 pseudo-utterances. The whisper task also witnesses a considerable PER reduction, even though less dramatic and more variable. Here, the best PER of 42.4% is reached for 600 pseudo-utterances. As discussed in Sect. 3.3, whisper recognition is arguably more challenging due to the higher confusability of phone classes in the acoustic space, which results in reduced ASR performance compared to the neutral task.

In the next step, we evaluate the impact of the number of mixture components in the utterance-generating

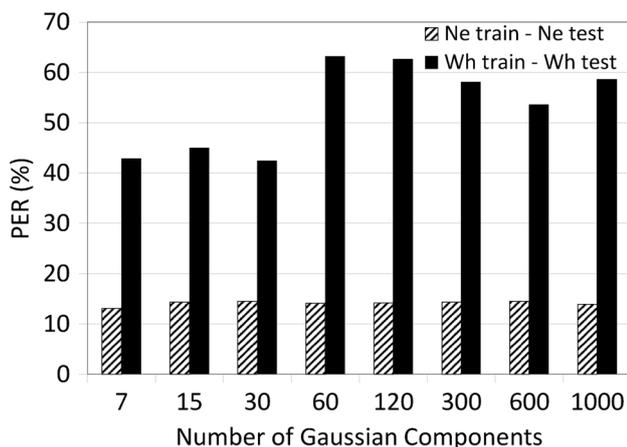


Fig. 4 DNN–HMM performance as a function of the number of Gaussian components in the GMM–UBM; PER (%). 300 pseudo-utterances included in the training set

GMM–UBM on the DNN–HMM performance (see Fig. 4). Here, the number of pseudo-utterances extending the original training set is fixed to 300. In the neutral task, the PERs show a steady, slightly increasing trend with the increasing number of mixture components. The lowest PER (13.1 %) is achieved with 7 components. The whisper task is more sensitive, with an abrupt PER increase between 30 and 60 components. The best PER (42.4 %) is provided by 30 mixtures. The trend differences between neutral and whispered data may be likely attributed to the different phonetic diversity in the neutral and whispered speech acoustic spaces. The authors hypothesize that the neutral acoustic-phonetic space is more spread and in spite of the limited size of the training data, can be successfully approximated by a higher number of Gaussians, while the whispered space is more monolithic and does not present a structure detailed enough to benefit from the higher number of Gaussians.

In summary, the experimental results in this section suggest that the proposed random sampling-based DNN training is successful in considerably reducing the DNN–HMM PERs compared to the baseline DNN–HMM trained only on the real samples. In addition, the new training strategy also helps to level or even surpass the performance of a GMM–HMM system which has access to the same original training data (GMM vs. DNN PERs: 27.5 vs. 13.1 % *Ne*; 45.0 vs. 42.4 % *Wh*). This suggests that random sampling-based DNN training has a good potential to bridge the gap between DNN–HMM and GMM–HMM systems in tasks where only sparse training data is available.

3.5 Temporal smoothing

As discussed in Sect. 2.2, random sampling from GMM–UBM produces feature vectors whose distributions approach those of the training data. However, after the samples are concatenated into pseudo-utterances, the resulting temporal trajectories are completely random. This will cause mismatch with the state transition models in the basic GMM–HMM used for the pseudo-utterance labeling (see Sect. 3.2, *preliminary stage*) and may also affect the efficiency of the LDA–MLLT transformations that were trained on concatenated real speech segments, and hence real speech trajectories. At the same time, the generated feature vectors still contain realistic first and second-order time derivatives from the GMM–UBM sampling and this temporal information will be meaningfully modeled by the HMM states. In this section, two approaches to temporal smoothing—RAS-TALP (Sect. 2.2.1) and *Frame-Shuffling* (Sect. 2.2.2) are experimentally studied.

Figures 5 and 6 show examples of the original and RASTALP-filtered cepstral trajectories (C_1 and C_2) in a real utterance and a pseudo-utterance, respectively. It can be seen that in both cases, the filter suppresses fast changes

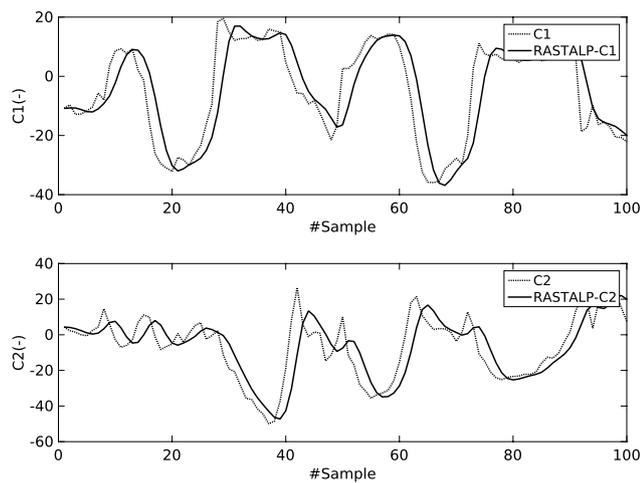


Fig. 5 Cepstral temporal trajectories in a real speech sample before and after RASTALP filtering

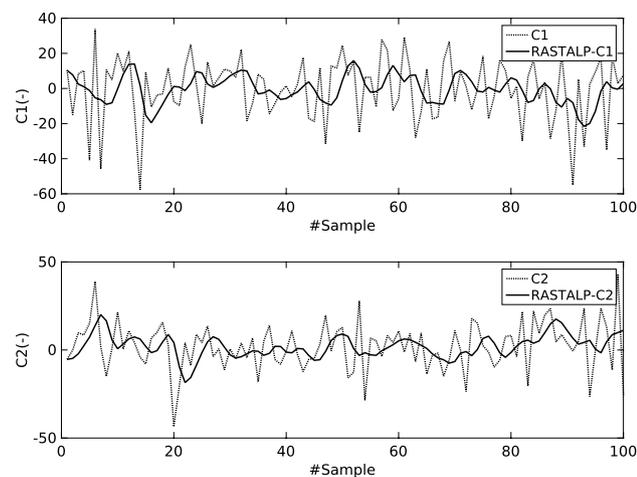


Fig. 6 Cepstral temporal trajectories in a pseudo-utterance before and after RASTALP filtering

in the temporal contours. As expected, cepstral tracks in the pseudo-utterance contain fast fluctuations and RASTALP is reducing their rate. At the same time, unlike in the real speech utterance, the filtered trajectories here considerably depart from the original ones, which may impact the frame feature distributions. As shown in Fig. 7, RASTALP has relatively small impact on the sample histograms in the real utterance (upper plots), while for the pseudo-utterance, the histograms before and after filtering differ significantly (center plots). This is an unfortunate artifact of linear filtering which will cause the RASTALP processed pseudo-utterances to depart from real speech in terms of frame vector distributions and in turn, affect the accuracy of the DNN–HMM state models. In the initial DNN–HMM experiment (following the setup from Sect. 3.4

with a 30-component GMM–UBM and 300 pseudo-utterances added to the training set), the DNN–HMM PER on the neutral task dropped from 13.1 % (without RASTALP) to 60.5%. In the RASTALP setup, the temporal filter was included in the feature extraction front-end and applied on all training and test data, including the GMM–UBM training stage. Clearly, the departure of the filtered pseudo-utterance distributions from those of real speech had a prevailing effect over the benefits of having less scattered temporal tracks.

In the next step, *Frame-Shuffling* (Sect. 2.2.2) is studied. The main purpose of this method is to reorder the pseudo-utterance frames so the Euclidean distances of the consecutive feature vectors would follow the ones seen in real speech, in terms of distance distributions. This is expected to result in more natural temporal contours while, unlike in RASTALP, preserving the frame distributions intact. The bottom plots in Fig. 7 capture cepstral histograms for the sample pseudo-utterance before and after *Frame-Shuffling*, which are clearly identical. Corresponding temporal cepstral trajectories before and after *Frame-Shuffling* are shown in Fig. 8. Here, the abrupt amplitude changes are notably reduced after shuffling, even though not to the extent seen in RASTALP.

Figure 9 presents between-frame distance histograms of real speech samples from the whole *Ne* set (upper plot), generated neutral pseudo-utterances (center plot), and shuffled neutral pseudo-utterances (bottom plot). The center plot confirms that the pseudo-utterance trajectories considerably depart from real speech in terms of variance. The bottom plot shows that *Frame-Shuffling* is successful in transforming the temporal variability towards real speech, in spite of introducing some artifacts in the resulting histogram due to the factors discussed in Sect. 2.2.2.

The effect of *Frame-Shuffling* on ASR is shown in Table 5. The PER on the neutral task is only slightly reduced, from 13.1 to 13.0%, and for whisper drops from 42.1 to 38.1%. Neither of these two gains are statistically significant (at a 95% confidence level). These feature analyses and ASR results show that *Frame-Shuffling* somewhat helps refining pseudo-utterances in a good direction, but at the same time also suggest that our concern about smoothing temporal trajectories to accommodate the state transition models in HMM might have been overstated. The DNN–HMM system in Sect. 3.4 trained on *unshuffled* pseudo-utterances was already well capable of outperforming its GMM–HMM counterpart. This implies that either the transition models were not considerably hurt by the exposure to the temporal randomness in the pseudo-utterances, or that the transition models have only a limited impact on the overall quality of an HMM. While the authors hypothesize that both factors are involved in some way, the latter hypothesis has been supported by studies that went as far as setting all HMM state transition probabilities to a constant,

Fig. 7 Cepstral histograms in a real speech sample before and after RASTALP and in a pseudo-utterance before and after applying RASTALP or *Frame-Shuffling*

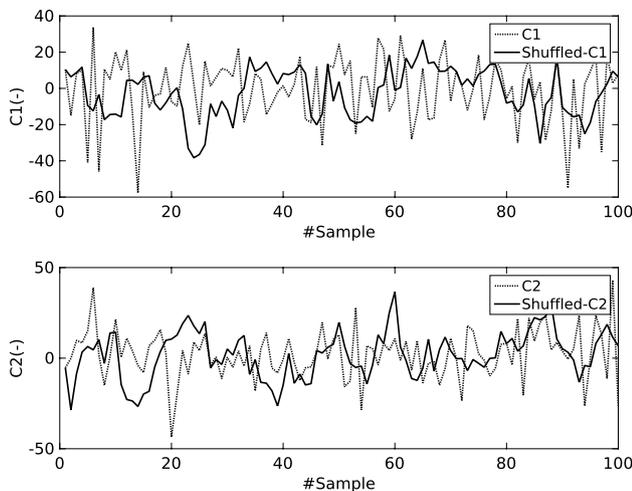
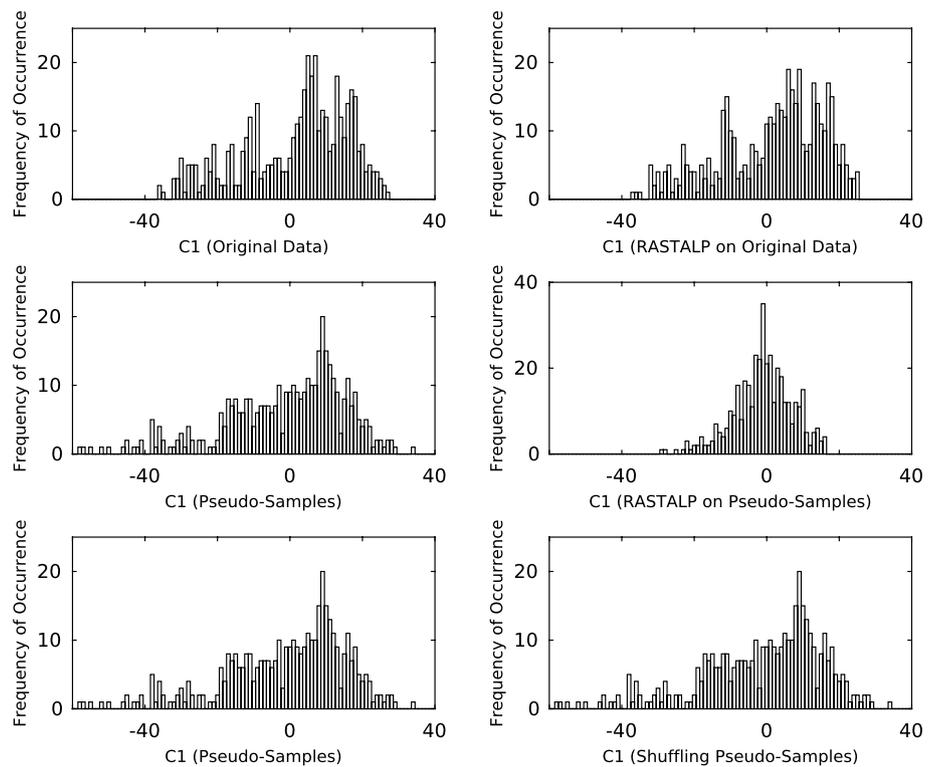


Fig. 8 Cepstral temporal trajectories in a pseudo-utterance before and after *Frame-Shuffling*

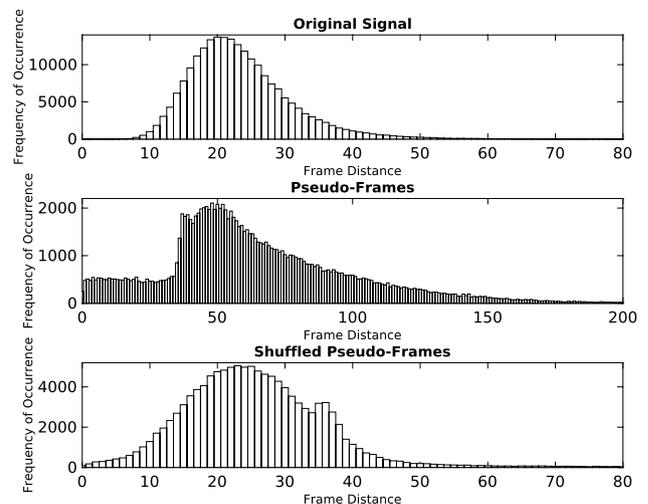


Fig. 9 Euclidean distance histograms in real speech data (neutral training set), pseudo-utterances, and shuffled pseudo-utterances

while attaining meaningful performance [e.g., Ketabdar and Boulard (2010)]. Finally, it is noted the pseudo-utterances still retain a good portion of the meaningful temporal information through the first- and second-order time derivatives included in the frame feature vectors, and this information is learned by the state models of the HMM system.

Table 5 Performance of DNN–HMM trained on pseudo-samples vs. shuffled pseudo-samples; PER (%)

Train	Test	Pseudo-samples	Shuffled pseudo-samples
Ne-VEII	Ne-VEII	13.1	13.0
Wh-VEII	Wh-VEII	42.4	38.1

4 Conclusions

DNN–HMM acoustic models have been increasingly popular in various speech-oriented applications, including speech recognition. Current DNN–HMM systems typically surpass traditional GMM–HMM based engines due to more efficient modeling of HMM state emission probabilities. However, the superior performance of DNN models comes at the cost of requiring more training data. There are numerous application domains where the available data is limited, yet sufficient for training/adapting meaningful GMM models, while being too sparse to train accurate DNNs. This is due to the inherently different nature of GMM and DNN modeling where GMMs, being parametric models, need access to modest sample sizes to generalize the distribution contours while DNNs rely on excessive examples to learn class boundaries.

The goal of this study was to bridge the gap between the DNN–HMM and GMM–HMM training when only sparse training samples are available. The proposed algorithm relies on a generative GMM–UBM model that will produce, through random sampling, an excessive number of pseudo-samples. The pseudo-samples effectively carry information about the source data space as learned by the GMM, and thanks to their quantity, this information can be successfully mediated to the DNN. Two methods were introduced—*random sampling of a GMM–UBM* which is used to generate pseudo-utterances for DNN training, and *Frame-Shuffling*, which reorders the pseudo-utterance frames to better match temporal trajectories of real speech. The proposed scheme was evaluated on a neutral and a whisper speech recognition task where only very limited training data were available (30 and 45 minutes, respectively). While the baseline GMM–HMM recognizer considerably outperformed the DNN–HMM setup, incorporating the proposed strategies in the DNN training helped the DNN–HMM to level and even surpass the GMM–HMM system. The evaluations showed 79.0 and 45.6% relative PER reduction from the baseline to the random-sampling trained DNN–HMM, on the neutral and whispered tasks. The random-sampling DNN–HMM provided a 52.7 and 15.3% relative PER reduction compared to the GMM–HMM system trained on the original data.

Funding This project was funded by AFRL under contract FA8750-12-1-0188 and partially by the University of Texas at Dallas from the Distinguished University Chair in Telecommunications Engineering held by J.H.L. Hansen.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Atal, B. S. (1974). Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. *The Journal of the Acoustical Society of America*, 55(6), 1304–1312.
- Bořil, H., Grézl, F., & Hansen, J. H. L. (2011). Front-end compensation methods for LVCSR under Lombard effect. *INTERSPEECH 2011* Florence, pp. 1257–1260.
- Bořil, H., & Hansen, J. H. L. (2011). UT-Scope: Towards LVCSR under Lombard effect induced by varying types and levels of noisy background. In *IEEE ICASSP 2011*, May 22–27, 2011, Prague, pp. 4472–4475.
- Bou-Ghazale, S., & Hansen, J. H. L. (1994). Duration and spectral based stress token generation for HMM speech recognition under stress. In *Proceedings of ICASSP '94*, Adelaide, April 19–22, pp. 413–416.
- Bou-Ghazale, S., & Hansen, J. H. L. (1996). Generating stressed speech from neutral speech using a modified celp vocoder. *Speech Communication*, 20(1–2), 93–110.
- Brooks, S. (1998). Markov chain monte carlo method and its application. *Journal of the Royal Statistical Society*, 47(1), 69–100.
- Casella, G., & George, E. I. (1992). Explaining the gibbs sampler. *The American Statistician*, 46(3), 167–174.
- Dahl, G., Yu, D., Deng, L., & Acero, A. (2012). Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1), 30–42.
- Davis, S. B., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4), 357–366.
- Deng, L., Hinton, G., & Kingsbury, B. New types of deep neural network learning for speech recognition and related applications: An overview. In *ICASSP 2013*, Vancouver, pp. 8599–8603.
- Gales, M. (1998). Maximum likelihood linear transformations for hmm-based speech recognition. *Computer Speech and Language*, 12, 75–98.
- Ghaffarzagdegan, S., Bořil, H., & Hansen, J. H. L. (2016). Generative modeling of pseudo-whisper for robust whispered speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(10), 1705–1720.
- Ghaffarzagdegan, S., Bořil, H., & Hansen, J. H. L. (2014a). Model and feature based compensation for whispered speech recognition. In *Interspeech 2014*, Singapore, pp. 2420–2424.
- Ghaffarzagdegan, S., Bořil, H., & Hansen, J. H. L. (2014b). UT-VOCAL EFFORT II: Analysis and constrained-lexicon recognition of whispered speech. In *IEEE ICASSP 2014*, Florence, pp. 2563–2567.
- Ghaffarzagdegan, S., Bořil, H., & Hansen, J. H. L. (2015). Generative modeling of pseudo-target domain adaptation samples for whispered speech recognition. In *IEEE ICASSP 2015*, Brisbane.
- Haeb-Umbach, R., & Ney, H. (1992). Linear discriminant analysis for improved large vocabulary continuous speech recognition. *ICASSP 1992*, Washington, DC, pp. 13–16.
- Hermansky, H., & Morgan, N. (1994). RASTA processing of speech. In *IEEE Transactions on Speech and Acoustics*, 2, 587–589.
- Hinton, G., Deng, L., Yu, D., rahman Mohamed, A., Jaitly, N., Senior, A., et al. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6), 82–97.
- Huang, C., & Moraga, C. (2004). A diffusion-neural-network for learning from small samples. *International Journal of Approximate Reasoning*, 35(2), 137–161.

- Ketabdar, H., & Boulard, H. (2010). Enhanced phone posteriors for improving speech recognition systems. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6), 1094–1106.
- Lasserre, J. A., Bishop, C. M., & Minka, T. P. (2006). Principled hybrids of generative and discriminative models. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1, CVPR '06*, Washington, DC, USA. IEEE Computer Society, pp. 87–94.
- Li, D., Wu, C., Tsai, T., & Lin, Y. (2007a). Using mega-trend-diffusion and artificial samples in small data set learning for early flexible manufacturing system scheduling knowledge. *Computers & Operations Research*, 34(4), 966–982.
- Li, D.-C., Hsu, H.-C., Tsai, T.-I., Lu, T.-J., & Hu, S. C. (2007b). A new method to help diagnose cancers for small sample size. *Expert Systems with Applications*, 33(2), 420–424.
- Liporace, L. (2006). Maximum likelihood estimation for multivariate observations of markov sources. *IEEE Transactions on Information Theory*, 28(5), 729–734.
- Mao, R., Chen, A., Zhang, L., & Zhu, H. (2006). A new method to assist small data set neural network learning. *2006 6th International Conference on Intelligent Systems Design and Applications*, 01:17–22.
- Matsoukas, S., Schwartz, R., Jin, H., & Nguyen, L. (1997). Practical implementations of speaker-adaptive training. In *DARPA Speech Recognition Workshop*.
- Neal, R. M. (2010). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54, 113–162.
- Ng, A. Y., & Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems 14*, MIT Press, pp. 841–848.
- Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., & Vesely, K. (2011). The Kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*.
- Rabiner, L. R. (1990). Readings in speech recognition. chapter A tutorial on hidden Markov models and selected applications in speech recognition, San Francisco: Morgan Kaufmann Publishers Inc., pp. 267–296.
- Rubinstein, Y. D. & Hastie, T. (1997). Discriminative vs informative learning. In *proc. third int. conf. on knowledge discovery and data mining*. AAAI Press, pp. 49–53
- Seide, F., Li, G., & Yu, D. (2011). Conversational speech transcription using context-dependent deep neural networks. International Speech Communication Association.
- Seltzer, M., Yu, D., & Wang, Y. (2013). An investigation of deep neural networks for noise robust speech recognition. In *IEEE ICASSP 2013*, Vancouver.
- Young, S. (1996). A review of large-vocabulary continuous-speech recognition. *IEEE Signal Processing Magazine*, 1996, 13(5).
- Zhang, C., & Hansen, J. H. L. (2009). Advancement in whisper-island detection with normally phonated audio streams. In *ISCA INTER-SPEECH*, Brighton, pp. 860–863.
- Zhang, X., Trmal, J., Povey, D., & Khudanpur, S. (2014). Improving deep neural network acoustic models using generalized max-out networks. In *ICASSP 2014*, Florence, May 4–9, 2014, pp. 215–219.
- Zue, V., Seneff, S., & Glass, J. (1990). Speech database development at MIT: TIMIT and beyond. *Speech Communication*, 9(4), 351–356.