공학박사학위논문

Deep Speaker Embedding for Robust Speaker Recognition

비화자 요소에 강인한 화자 인식을 위한 딥러닝 기반 성문 추출

2021년 2월

서울대학교 대학원

전기 · 컴퓨터공학부

강 우 현

Deep speaker embedding for robust speaker recognition

비화자 요소에 강인한 화자 인식을 위한
딥러닝 기반 성문 추출

지도교수 김 남 수

이 논문을 공학박사 학위논문으로 제출함
2021년 1월

서울대학교 대학원
전기 · 컴퓨터공학부
강 우 현

강우현의 공학박사 학위논문을 인준함
2020년 12월

위 원 장	김 성 철	[illegible] (인)
부위원장	김 남 수	[illegible] (인)
위 원	심 병 효	[illegible] (인)
위 원	장 준 혁	[illegible] (인)
위 원	신 종 원	[illegible] (인)

Abstract

Over the recent years, various deep learning-based embedding methods have been proposed and have shown impressive performance in speaker verification. However, as in most of the classical embedding techniques, the deep learning-based methods are known to suffer from severe performance degradation when dealing with speech samples with different conditions (e.g., recording devices, emotional states). Also, unlike the classical Gaussian mixture model (GMM)-based techniques (e.g., GMM supervector or i-vector), since the deep learning-based embedding systems are trained in a fully supervised manner, it is impossible for them to utilize unlabeled dataset when training.

In this thesis, we propose a variational autoencoder (VAE)-based embedding framework, which extracts the total variability embedding and a representation for the uncertainty within the input speech distribution. Unlike the conventional deep learning-based embedding techniques (e.g., d-vector or x-vector), the proposed VAE-based embedding system is trained in an unsupervised manner, which enables the utilization of unlabeled datasets. Furthermore, in order to prevent the potential loss of information caused by the Kullback-Leibler divergence regularization term in the VAE-based embedding framework, we propose an adversarially learned inference (ALI)-based embedding technique. Both VAE- and ALI-based embedding techniques have shown great performance in terms of short duration speaker verification, outperforming the conventional i-vector framework.

Additionally, we present a fully supervised training method for disentangling the non-speaker nuisance information from the speaker embedding. The proposed training scheme jointly extracts the speaker and nuisance attribute (e.g., recording channel, emotion) embeddings, and train them to have maximum information on their main-task while ensuring maximum uncertainty on their sub-task. Since the proposed method does not require any heuristic training strategy as in the conventional disentanglement techniques (e.g., adversarial learning, gradient reversal), optimizing the embedding network is relatively more stable. The proposed scheme have shown state-of-the-art performance in RSR2015 Part 3 dataset, and demonstrated its capability in efficiently disentangling the recording device and emotional information from the speaker embedding.

Keywords: Robust speaker recognition, speech embedding, speaker verification, unsupervised representation learning, supervised disentanglement.

Student number: 2014-21697

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Speaker recognition is the process of recognizing a user's identity using the characteristics extracted from their voices, which can be classified into two tasks: speaker identification and verification. More specifically, speaker verification is a task of verifying the claimed speaker identity given the voice sample. As depicted in Figure 1.1, the speaker verification process usually consists of three stages: acoustic feature extraction, utterance-level feature modeling, and decision. The acoustic feature extraction stage analyzes the speech in a short duration frame. In practice, a mel frequency cepstral coefficient (MFCC) or spectrogram feature is extracted from each frame to capture the instantaneous vocal tract characteristics. The utterance-level feature extraction, or embedding stage summarizes the frame-level information into a fixed dimension vector. The extracted utterance-level feature, or embedding vector represents the overall pattern of the speaker's vocal tract shape. Finally, the decision stage measures the similarity between a given pair of embedding vectors.

Many previous studies on embedding focused on efficiently reducing the dimensionality of the Gaussian mixture model (GMM) supervector, which is a concate-

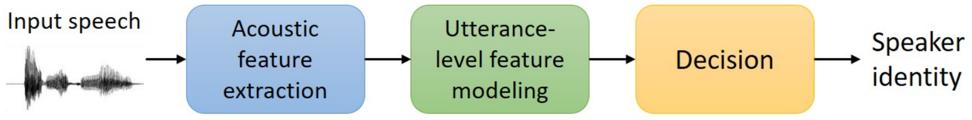


Figure 1.1: Basic process of speaker verification.

nation of the mean vectors of each mixture component [1], while preserving the speaker relevant information via factorization (e.g., eigenvoice adaptation and joint factor analysis) [2, 3]. Particularly the i-vector framework [4, 5], which projects the variability within the GMM supervector caused by various factors (e.g., channel and speaker) onto a low dimensional subspace, has become one of the most dominant techniques used in speaker recognition. The i-vector framework is essentially a linear factorization technique which decomposes the variability of the GMM supervector into a total variability matrix and an i-vector (i.e. total factor). Due to its linear nature, the i-vector, along with most of the other utterance-level representations driven via factorization, is not considered to fully capture the whole variability of the given speech utterances.

In recent years, various methods have been proposed utilizing deep learning architectures for extracting embedding vectors and have shown better performance than the i-vector framework when a large amount of training data is available [6]. In [7], a deep neural network (DNN) for frame-level speaker identification was trained and the averaged activation from the last hidden layer, namely, the d-vector, was taken as the embedding vector for text-dependent speaker verification. In [6, 8], a speaker identification model consisting of a frame-level network and a segment-level network was trained and the hidden layer activation of the segment-level network (i.e. x-vector) was extracted as the embedding vector. In [9], long short-term memory (LSTM) layers were adopted to capture the contextual information within the d-vector, and the embedding network was trained to directly optimize the verification score (e.g., cosine similarity) in an end-to-end fashion. The end-to-end d-vector framework was further enhanced in [10] by applying different weight (i.e. attention) to each frame-level activation while obtaining the d-vector, which enables the embedding network to attend more on the frames with relatively higher amount of speaker-dependent information. In [11], a generalized end-to-end loss function, which optimizes the embedding vector to move towards the centroid of the true speaker while departing away from the centroid of the most confusing speaker, was introduced to train the end-to-end d-vector system more efficiently. In [12] and [13], a variational autoencoder (VAE)-based architecture was trained in an unsupervised manner to extract an embedding vector for short-duration speaker verification. Despite their success in well-matched conditions, the deep learning-based embedding methods are vulnerable to the performance degradation caused by mismatched conditions (e.g., channel, noise) [14]. Moreover, since most of the previously proposed deep learning-based feature extraction models are trained in a supervised manner, it is impossible to use them when little to no labeled data is available for training.

In real life applications, numerous factors can contribute to the mismatches in speaker verification [15]. Especially in forensic situations, channel mismatch often occurs since police officers usually acquire voice recordings using various recording devices (e.g., hidden microphones, mobile phones) [16]. Such variation in recording devices is known to cause variability to the speech distribution, which leads to low speaker identification or verification performance.

Furthermore, due to the increasing demand for voice-based authentication systems, verifying users with randomized pass-phrase with constrained vocabulary has become an important task [17]. This particular task is called the random digit strings speaker verification, where the speakers are enrolled and tested with random sequences of digits. The random digit strings task highlights one of the most serious causes for feature uncertainty, which is the short duration of the given speech samples [18]. The conventional i-vector is known to suffer from severe performance degradation when short duration speech is applied to the verification process [19]. It has been reported that the i-vectors extracted from short duration speech samples are relatively unstable [19, 20, 16]. The short duration problem can be critical when it comes to real life applications since, in most practical systems, the speech recording for enrollment and trial is required to be short.

This dissertation proposes several embedding methods to tackle the following issues:

- unsupervised deep learning based embedding techniques for short durational speaker verification,
- supervised training method for disentangling the nuisance attribute (e.g., recording channel) information from the speaker embedding.

In Chapter 3, we propose a novel technique for extracting an i-vector-like feature based on the variational autoencoder (VAE) which summarizes the variability within the Gaussian mixture model (GMM) distribution through a non-linear process. Analogous to the conventional i-vector framework, the proposed VAE is trained to generate the GMM supervector according to the maximum likelihood criterion given the Baum-Welch statistics of the input utterance. The proposed framework is compared with the conventional i-vector method using the TIDIGITS dataset. Experimental results show that the proposed method can cope with the uncertainty caused by the short duration.

In Chapter 4, we propose a novel technique for extracting an i-vector-like feature based on an adversarially learned inference (ALI) model which summarizes the variability within the Gaussian mixture model (GMM) distribution through a non-linear process. Analogous to the VAE-based feature extractor, the proposed ALI-based model is trained to generate the GMM supervector according to the maximum likelihood criterion given the Baum-Welch statistics of the input utterance. However, in order to prevent the potential loss of information caused by the Kullback-Leibler divergence (KL divergence) regularization term in the training criterion of the VAE-based model, the newly proposed ALI-based feature extractor exploits a joint discriminator to ensure that the generated latent variable and the GMM supervector are realistic. The proposed framework is compared with the conventional i-vector and VAE-based methods using the TIDIGITS dataset. Experimental results show that the proposed method can represent the uncertainty caused by the short duration better than the VAE-based method. Furthermore, the proposed approach has shown great performance when applied in association with the standard i-vector framework.

In Chapter 5, we propose a novel fully supervised training method for extracting a speaker embedding vector disentangled from the variability caused by the nuisance attributes. The proposed framework was compared with the conventional deep learning-based embedding methods using the RSR2015 and VoxCeleb1 dataset. Experimental results show that the proposed approach can extract speaker embeddings robust to channel and emotional variability.

The rest of this thesis is organized as follows: The next chapter introduces the conventional embedding techniques (e.g., i-vector, d-vector, and x-vector) and disentangling methods (e.g., gradient reversal, adversarial training strategy). In Chapter 3, we propose a VAE-based total variability embedding method for short duration speaker verification. In Chapter 4, we propose a ALI-based total variability embedding method which replaces the KL-divergence regularization of the VAE-based embedding framework with a adversarial training strategy. In Chapter 5, we propose the joint factor embedding (JFE) framework for disentangling the nuisance attribute information from the speaker embedding. Finally, conclusions are drawn in Chapter 6.

Chapter 2

Conventional embedding techniques for speaker recognition

2.1 i-vector Framework

Once a universal background model (UBM) is provided, which is a GMM representing the utterance-independent distribution of the frame-level features, an utterance-dependent GMM can be attained by adjusting the UBM parameters using the maximum a posteriori (MAP) adaptation algorithm [21]. By concatenating the mean vector of each Gaussian mixture component, a GMM supervector can be obtained, which summarizes the overall pattern of the frame-level feature distribution. However, using the GMM supervector as an utterance-level feature may limit the overall speaker recognition performance due to its high dimensionality.

To solve this problem, various methods for reducing the dimensionality of the

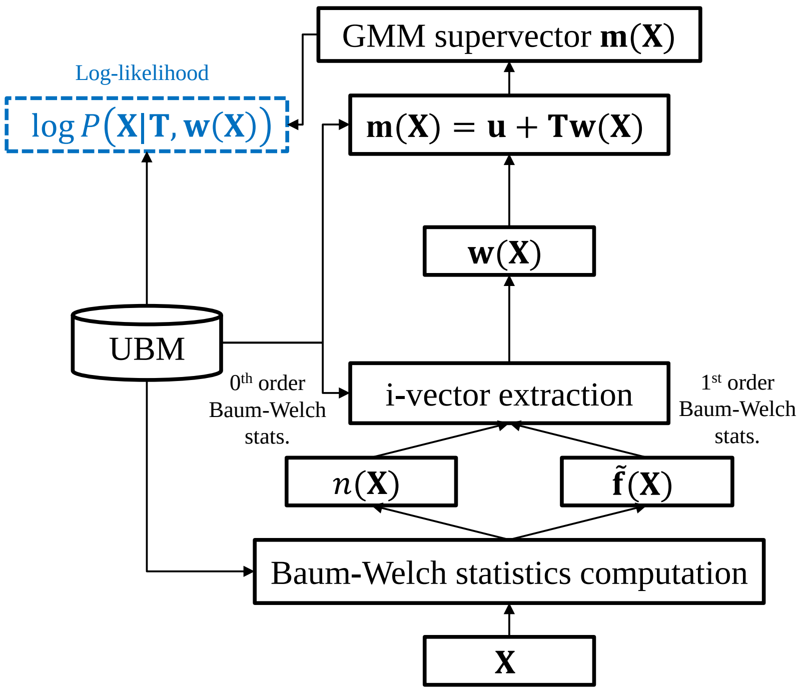

GMM supervector $\mathbf{m}(\mathbf{X})$
Log-likelihood
$\log P(\mathbf{X}|\mathbf{T}, \mathbf{w}(\mathbf{X}))$
$\mathbf{m}(\mathbf{X}) = \mathbf{u} + \mathbf{T}\mathbf{w}(\mathbf{X})$
$\mathbf{w}(\mathbf{X})$
UBM
i-vector extraction
0th order Baum-Welch stats.
1st order Baum-Welch stats.
$n(\mathbf{X})$
$\tilde{\mathbf{f}}(\mathbf{X})$
Baum-Welch statistics computation
$\mathbf{X}$

Figure 2.1: Flow chart of the i-vector framework.

GMM supervector have been proposed [2, 3, 4]. Specifically, the i-vector framework is now widely used to represent the idiosyncratic characteristics of the utterance in speaker and language recognition [22]. The extraction of an i-vector can be viewed as a factorization process decomposing the GMM supervector as

$$\mathbf{m}(\mathbf{X}) = \mathbf{u} + \mathbf{T}\mathbf{w}(\mathbf{X}) \tag{2.1}$$

where $\mathbf{m}(\mathbf{X})$, $\mathbf{u}$, $\mathbf{T}$, and $\mathbf{w}(\mathbf{X})$ indicate the ideal GMM supervector dependent on a given speech utterance $\mathbf{X}$, UBM supervector, total variability matrix, and i-vector, respectively. The i-vector framework aims to find the optimal $\mathbf{w}(\mathbf{X})$ and $\mathbf{T}$ to adapt the UBM parameters to a given speech utterance $\mathbf{X}$. Given $\mathbf{X}$, the zeroth- and the first-order Baum-Welch statistics are obtained as

$$n_c(\mathbf{X}) = \sum_{l=1}^{L} \gamma_l(c) \tag{2.2}$$

$$\tilde{\mathbf{f}}_c(\mathbf{X}) = \sum_{l=1}^{L} \gamma_l(c)(\mathbf{x}_l - \mathbf{u}_c) \tag{2.3}$$

where for each frame l within $\mathbf{X}$ with L frames, $\gamma_l(c)$ is the posterior probability that the l^{th} frame feature $\mathbf{x}_l$ is aligned to the c^{th} mixture component of the UBM, $\mathbf{u}_c$ is the mean vector of the c^{th} mixture component of the UBM, and $n_c(\mathbf{X})$ and $\tilde{\mathbf{f}}_c(\mathbf{X})$ are the zeroth- and the centralized first-order Baum-Welch statistics, respectively.

As shown in Fig. 2.1, the i-vector framework can be considered as an adaptation process where the mean of each UBM mixture component is adjusted to maximize the likelihood with respect to a given utterance, and the estimated i-vector is served as the adaptation factor. Let $\mathbf{\Sigma}_c$ denote the covariance matrix of the c^{th} UBM mixture component and F be the dimensionality of the frame-level features. The log-likelihood given an utterance $\mathbf{X}$ conditioned on $\mathbf{w}(\mathbf{X})$ can be obtained as

$$\begin{aligned}\log P(\mathbf{X}|\mathbf{T}, \mathbf{w}(\mathbf{X})) = &\sum_{c=1}^{C}(n_c(\mathbf{X}) \log \frac{1}{(2\pi)^{F/2}|\mathbf{\Sigma}_c|^{1/2}} \\ &- \frac{1}{2}\sum_{l=1}^{L} \gamma_l(c)(\mathbf{x}_l - \mathbf{m}_c(\mathbf{X}))^t \mathbf{\Sigma}_c^{-1}(\mathbf{x}_l - \mathbf{m}_c(\mathbf{X}))),\end{aligned} \tag{2.4}$$

where $\mathbf{m}_c(\mathbf{X})$ is the mean of the c^{th} mixture component of $\mathbf{m}(\mathbf{X})$ and the superscript t indicates matrix transpose. The log-likelihood given $\mathbf{X}$ is obtained by marginalizing (4) over $\mathbf{w}(X)$ as

$$\begin{aligned}\log P(\mathbf{X}|\mathbf{T}) =& \log \mathbb{E}_{\mathbf{w}}[P(\mathbf{X}|\mathbf{T}, \mathbf{w})] \\ =& \log \int P(\mathbf{X}|\mathbf{T}, \mathbf{w}) \mathcal{N}(\mathbf{w}|0, \mathbf{I}) d\mathbf{w}.\end{aligned} \tag{2.5}$$

The total variability matrix $\mathbf{T}$ is trained to maximize the log-likelihood (5) via the expectation-maximization (EM) algorithm. Interested readers are encouraged to refer to [4] and [5] for further details of the i-vector framework.

2.2 Deep learning-based speaker embedding

2.2.1 Deep embedding network

Two of the most widely used speaker embedding techniques are the LSTM-based d-vector [9] and the TDNN (time-delay DNN)-based x-vector system [6]. In both frameworks, given a speech utterance $\mathbf{X}$ with T frames, a sequence of frame-level acoustic features $\{\mathbf{x}_1, ..., \mathbf{x}_T\}$ extracted from $\mathbf{X}$ is fed into the frame-level network. In the d-vector system, one of most widely used technique for text-dependent speaker recognition, the frame-level network is composed of LSTM layers, which helps capture the temporal correlation. On the other hand, the frame-level network of the x-vector system consists of TDNN layers, which is often used for text-independent speaker recognition. Once the frame-level outputs $\{\mathbf{h}_1, ..., \mathbf{h}_T\}$ are obtained, they are aggregated to obtain an utterance-level representation. One way of aggregating the frame-level outputs is to compute the weighted average as

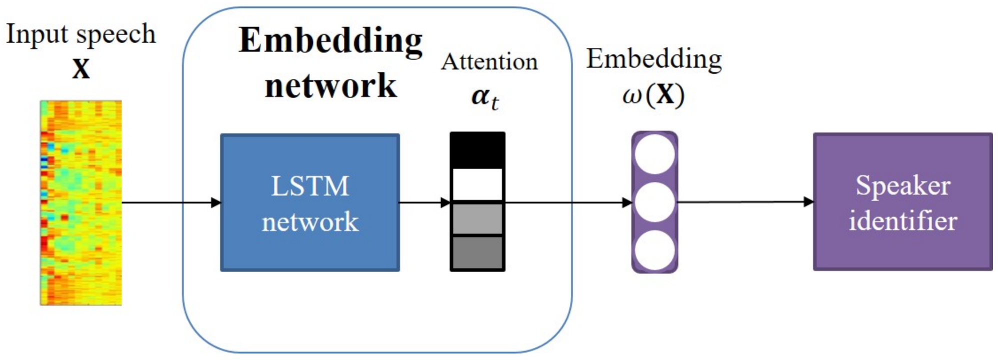
$$\omega = \sum_{t=1}^{T} \alpha_t \mathbf{h}_t \tag{2.6}$$

where $\alpha_t \in [0, 1]$ is a normalized weight, which is computed by

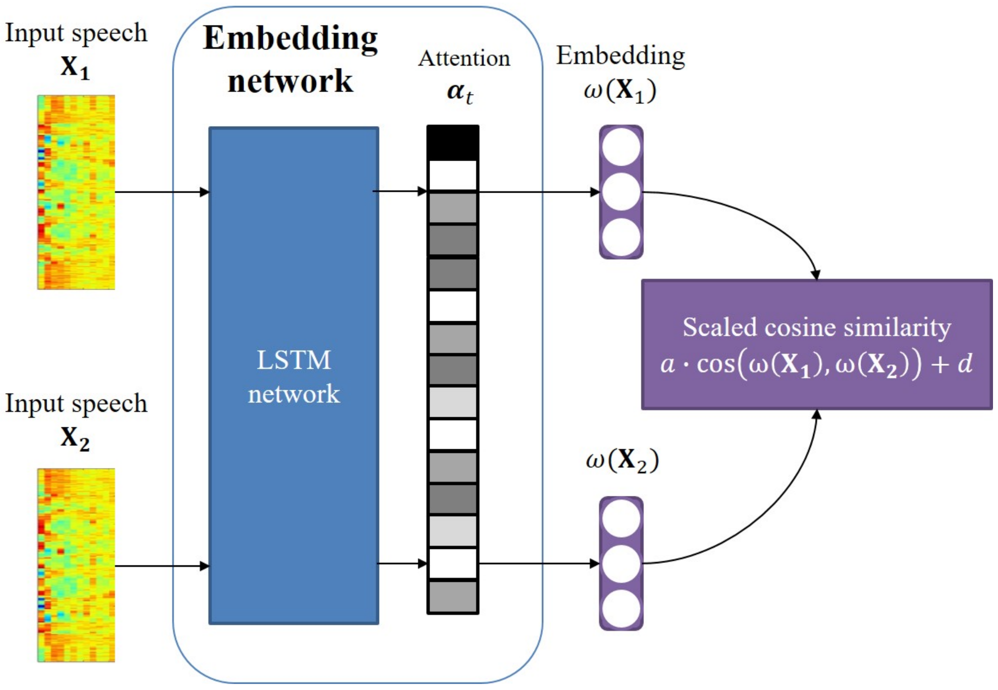
$$\alpha_t = \frac{\exp(e_t)}{\sum_{t=1}^{T} \exp(e_t)}. \tag{2.7}$$

In (2.7), the frame-level score (i.e. attention) e_t is computed as follows:

$$e_t = \mathbf{v}_t^{\intercal} \tanh(\mathbf{W}_t \mathbf{h}_t + \mathbf{b}_t) \tag{2.8}$$



(a)



(b)

Figure 2.2: (a) LSTM-based d-vector system trained with softmax loss. (b) LSTM-based d-vector system trained with end-to-end loss.

where $\mathbf{v}_t$, $\mathbf{W}_t$, and $\mathbf{b}_t$ are trainable parameters and superscript $\intercal$ indicates transpose operation. By using different weight for each frame, speech frames with relatively higher speaker-relevancy can contribute more to the embedding vector.

The embedding network is trained by either minimizing the speaker identification loss [7] or directly optimizing the verification performance (i.e. end-to-end speaker verification) [10, 11]. In the first case (i.e. embedding network trained for identification), as shown in Fig. 2.2a, a feed-forward neural network for classifying the speakers in the training set is trained jointly with the embedding network. The speaker classification network takes the utterance-level representation ω as input and has an N-dimensional softmax output $\tilde{\mathbf{y}}(\omega)$ where N corresponds to the number of training speakers. Given the one-hot speaker label $\mathbf{y}$, the embedding and classification networks are trained to minimize the following cross-entropy loss function:

$$\mathbf{L}_{spkr} = -\sum_{n=1}^{N} \mathbf{y}_n \log \tilde{\mathbf{y}}_n(\omega) \tag{2.9}$$

where $\mathbf{y}_n$ and $\tilde{\mathbf{y}}_n(\omega)$ are the n^{th} components of $\mathbf{y}$ and $\tilde{\mathbf{y}}(\omega)$, respectively.

For training the end-to-end speaker verification system (i.e. embedding network trained for verification), a mini-batch of $J \times K$ utterances is fed into the embedding network where the mini-batch is composed of J speakers, and each speaker has K utterances. As depicted in Fig. 2.2b, the scaled cosine similarity between each embedding vector and the centroid of the embedding vectors from each speaker are computed by

$$\mathbf{S}_{jk,i} = a \cdot \cos(\omega_{jk}, \mathbf{c}_i) + d \tag{2.10}$$

where a and d are trainable parameters, and $\cos(\omega_{jk}, \mathbf{c}_i)$ is the cosine similarity between the utterance-level representation extracted from the k^{th} utterance of the j^{th} speaker ω_{jk} and the centroid of the i^{th} speaker's utterance-level representations

$\mathbf{c}_i$ ($1 \leq j, i \leq J$ and $1 \leq k \leq K$). For each utterance-level representation ω_{jk} in the mini-batch, the embedding network is trained to maximize the following end-to-end loss function:

$$\mathbf{L}_{e2e} = \mathbf{S}_{jk,j} - \log \sum_{i=1, i \neq j}^{J} \exp(\mathbf{S}_{jk,i}). \tag{2.11}$$

The end-to-end system is known to outperform the softmax method when a large amount of dataset is used for training [8], [10].

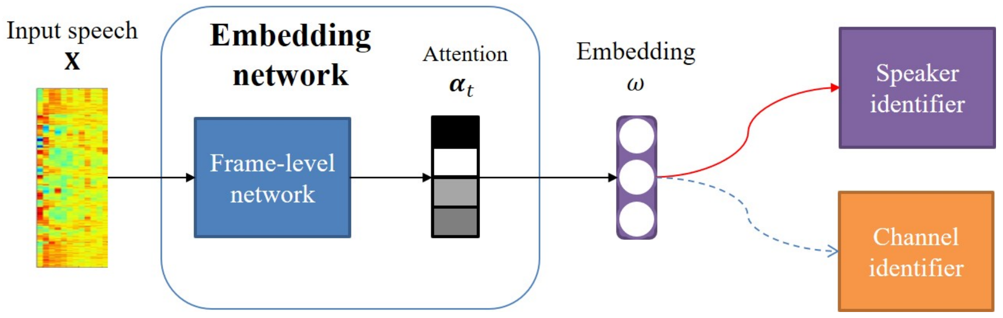
Once the embedding network is trained, the utterance-level representation ω [9], or the hidden layer activation of the speaker classification network [6] can be used as the speaker embedding vector.

2.2.2 Conventional disentanglement methods

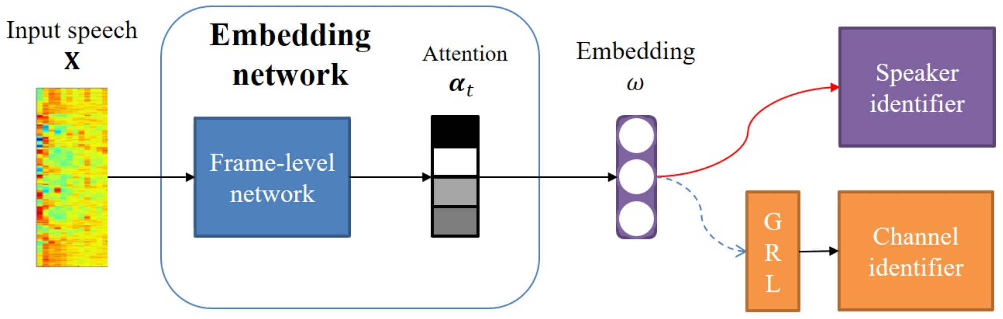
Recently, disentangling various non-speaker factors (e.g., channel type, noise type, noise-level) from the embedding vector has become an important issue in speaker verification [14, 23, 24]. Most of the techniques developed to address this issue are based on the multi-task learning (MTL) approaches [25] where the embedding network is trained to optimize in two tasks: main task (i.e. speaker classification) and subtask (e.g., channel classification) as shown in Fig. 2.3a. The objective of the MTL-based disentanglement technique is to achieve the best performance in the main task while degrading the performance in the subtask.

Gradient reversal strategy

One way to achieve this is the gradient reversal strategy, which has shown meaningful performance in channel-robust [23] and noise-robust [14] speaker verification. As shown in Fig. 2.3b, the gradient reversal strategy adds a gradient reversal layer



(a)



(b)

Figure 2.3: (a) Standard multi-task learning (MTL) architecture. (b) Domain adversarial training via gradient reversal layer (GRL).

(GRL) [26] between the subtask network and the embedding network. Let θ_{emb}, θ_{main}, θ_{sub} denote the parameters for the embedding, main task, and subtask networks. The GRL performs identity transformation on the input during forward propagation and reverses the gradient by multiplying a negative scalar $-\lambda$ during back-propagation. When jointly training the networks, the parameters are updated as

$$\theta_{emb} \leftarrow \theta_{emb} - l \cdot (\frac{\partial \mathbf{L}_{main}}{\partial \theta_{emb}} - \lambda \frac{\partial \mathbf{L}_{sub}}{\partial \theta_{emb}}), \tag{2.12}$$

$$\theta_{main} \leftarrow \theta_{main} - l \cdot (\frac{\partial \mathbf{L}_{main}}{\partial \theta_{main}}), \tag{2.13}$$

$$\theta_{sub} \leftarrow \theta_{sub} - l \cdot (\frac{\partial \mathbf{L}_{sub}}{\partial \theta_{sub}}) \tag{2.14}$$

where l, $\mathbf{L}_{main}$, and $\mathbf{L}_{sub}$ are the learning rate, loss functions for the main task and subtask, respectively. For extracting a channel-robust embedding for speaker verification, $\mathbf{L}_{main}$ would be the speaker cross-entropy $\mathbf{L}_{spkr}$ defined in (2.9), and $\mathbf{L}_{sub}$ would be the channel cross-entropy which can be computed as follows:

$$\mathbf{L}_{chan} = -\sum_{m=1}^{M} \mathbf{r}_m \log \tilde{\mathbf{r}}_m(\omega) \tag{2.15}$$

where M is the number of different channels (e.g., recording devices) in the training set, $\mathbf{r}_m$ and $\tilde{\mathbf{r}}_m(\omega)$ are the m^{th} component of the one-hot channel label $\mathbf{r}$ and channel classifier's softmax output $\tilde{\mathbf{r}}(\omega)$, respectively.

Anti-loss strategy

Another way to achieve disentanglement is by training the embedding network and the subtask network in a competitive manner via adversarial training [24]. The subtask network is trained to classify the channel identity correctly given the embedding vector as in (2.15). On the other hand, the main task and embedding networks are trained to discriminate the speaker by minimizing (2.9) but not to perform well on the subtask. In order to ensure high uncertainty on the subtask, [24] introduces anti-label when computing the cross-entropy for the subtask. The anti-label is obtained by flipping each bit in the one-hot label vector. This indicates that for channel disentanglement, the anti-loss can be computed as follows:

$$\mathbf{L}_{anti-dev} = -\sum_{m=1}^{M}(1-\mathbf{r}_m)\log\tilde{\mathbf{r}}_m(\omega). \tag{2.16}$$

By minimizing $\mathbf{L}_{anti-dev}$ and $\mathbf{L}_{speaker}$ simultaneously, the embedding network would be trained to produce a speaker discriminative embedding vector which is robust to channel variability.

Chapter 3

Unsupervised learning of total variability embedding for speaker verification with random digit strings

3.1 Introduction

Many previous studies on utterance-level features focused on efficiently reducing the dimensionality of the Gaussian mixture model (GMM) supervector, which is a concatenation of the mean vectors of each mixture component [1], while preserving the speaker relevant information via factorization (e.g., eigenvoice adaptation and joint factor analysis) [2, 3]. Particularly the i-vector framework [4, 5], which projects the variability within the GMM supervector caused by various factors (e.g., channel and speaker) onto a low dimensional subspace, has become one of the most dominant techniques used in speaker recognition. The i-vector framework is essentially a linear factorization technique which decomposes the variability of the GMM supervector into a total variability matrix and an i-vector (i.e. total factor). Due to its linear nature, the i-vector, along with most of the other utterance-level representations driven via factorization, is not considered to fully capture the whole variability of the given speech utterances.

Recently, various studies are being carried out for non-linearly extracting utterance-level features via deep learning. In [7], a DNN for frame-level speaker classification is trained and the activations of the last hidden layer, namely the d-vectors, are taken as a non-linear speaker representation. In [8] and [6], a TDNN-based utterance embedding technique is proposed, where the embedding is obtained by statistically pooling the frame-level activations of the TDNN. In [10] and [11], the speaker embedding neural networks are optimized to directly minimize the verification loss in an end-to-end fashion. However, since most of the previously proposed deep learning-based feature extraction models are trained in a supervised manner, it is impossible to use them when little to no labeled data is available for training.

Nowadays, due to the increasing demand for voice-based authentication systems, verifying users with randomized pass-phrase with constrained vocabulary has become an important task [17]. This particular task is called the random digit strings speaker verification, where the speakers are enrolled and tested with random sequences of digits. The random digit strings task highlights one of the most serious causes for feature uncertainty, which is the short duration of the given speech samples [18]. The conventional i-vector is known to suffer from severe performance degradation when short duration speech is applied to the verification process [19]. It has been reported that the i-vectors extracted from short duration speech samples are relatively unstable [19, 20, 16]. The short duration problem can be critical when it comes to real life applications since, in most practical systems, the speech recording for enrollment and trial is required to be short.

In this paper, we propose a novel approach to speech embedding for speaker recognition. The proposed method employs a variational inference model inspired by the variational autoencoder (VAE) [27, 28] to capture the total variability of the speech in a non-linear fashion. The VAE has an autoencoder-like architecture which assumes that the data is generated through a directed graphical model driven by a random latent variable. Analogous to the conventional i-vector adaptation scheme, the proposed model is trained according to the maximum likelihood criterion given the input speech. By using the mean and the variance of the latent variable as the utterance-level features, the proposed system is expected to take the uncertainty caused by short duration utterances into account. In contrast to the conventional deep learning-based feature extraction techniques, the proposed approach exploits the resources used in the conventional i-vector scheme (e.g., universal background model and Baum-Welch statistics) and remaps the relationship between the total factor and the total variability subspace through a non-linear process. Therefore the proposed feature extractor can substitute the conventional i-vector extraction module without any difficulty. Furthermore, since the proposed feature extractor is trained in an unsupervised fashion, no phonetic or speaker label is required for training.

In order to evaluate the performance of the unsupervised embedding techniques in the random digits task, we conducted a set of experiments using the TIDIG-ITS dataset. Experimental results show that the proposed method outperforms the standard i-vector framework in terms of equal error rate (EER), classification error, and decision cost function (DCF) measurements. It is also interesting to see that a dramatic performance improvement is observed when the features extracted from the proposed method and the conventional i-vector are augmented together. This indicates that the newly proposed feature and the conventional i-vector are complementary with each other.

3.2 Variational autoencoder

VAE is a variant of an autoencoder aiming to reconstruct the input at the output layer [27]. The main difference between VAE and an ordinary autoencoder is that the former assumes that the observed data $\mathbf{x}$ is generated from a random latent variable $\mathbf{z}$ which has a specific prior distribution such as the standard Gaussian. The VAE is composed of two directed networks: encoder and decoder networks. The encoder network outputs the mean and variance of the posterior distribution $p(\mathbf{z}|\mathbf{x})$ given an observation $\mathbf{x}$. Using the latent variable distribution generated by the encoder network, the decoder network tries to reconstruct the input pattern of the VAE at the output layer.

Given a training sample $\mathbf{x}$, the VAE aims to maximize the log-likelihood which can be written as follows [27]:

$$\mathrm{log} p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) + \mathscr{L}(\theta, \phi; \mathbf{x}). \tag{3.1}$$

In (3.1), ϕ denotes the variational parameters and θ represents the generative parameters. The first term in the right hand side (RHS) of (3.1) means the Kullback-Leibler divergence (KL divergence) between the approximated posterior $q_\phi(\mathbf{z}|\mathbf{x})$ and the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$ of the latent variable, which measures the dissimilarity between these two distributions. Since the KL divergence has a non-negative value, the second term in the RHS of (3.1) becomes the variational lower bound on the log-likelihood, which can be written as

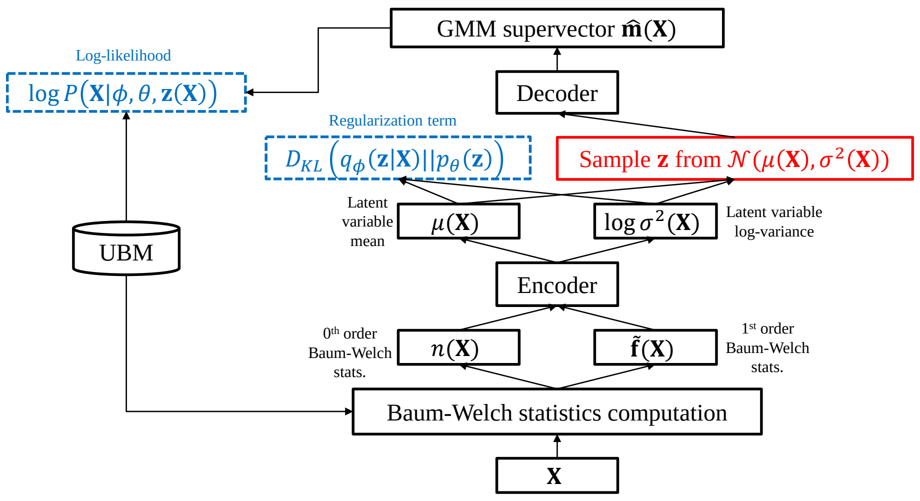
$$\begin{aligned} \log p_\theta(\mathbf{x}) \geq & \mathscr{L}(\theta, \phi; \mathbf{x}) \\ = & - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) \\ & + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \end{aligned} \tag{3.2}$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ are respectively specified by the encoder and decoder networks of the VAE.

The encoder and the decoder networks of the VAE can be trained jointly by maximizing the variational lower bound, which is equivalent to minimizing the following objective function [29]:

$$\begin{aligned} E_{VAE}(\mathbf{x}) = & D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) \\ & - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]. \end{aligned} \tag{3.3}$$

The first term in the RHS of (3.3) implies the KL divergence between the prior distribution and the posterior distribution of the latent variable $\mathbf{z}$, which regularizes the encoder parameters. On the other hand, the second term can be interpreted as the reconstruction error between the input and output of the VAE. Thus the VAE is trained not only to minimize the reconstruction error but also to maximize the similarity between the prior and posterior distributions of the latent variable.



GMM supervector $\hat{\mathbf{m}}(\mathbf{X})$
Log-likelihood
$\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))$
Decoder
Regularization term
$D_{KL}\left(q_\phi(\mathbf{z}|\mathbf{X})||p_\theta(\mathbf{z})\right)$
Sample $\mathbf{z}$ from $\mathcal{N}(\mu(\mathbf{X}),\sigma^2(\mathbf{X}))$
Latent variable mean
$\mu(\mathbf{X})$
$\log\sigma^2(\mathbf{X})$
Latent variable log-variance
UBM
Encoder
0th order Baum-Welch stats.
$n(\mathbf{X})$
$\tilde{\mathbf{f}}(\mathbf{X})$
1st order Baum-Welch stats.
Baum-Welch statistics computation
$\mathbf{X}$

Figure 3.1: Proposed VAE for non-linear feature extraction. Blue shows the loss terms. Red shows the sampling operations.

3.3 Variational inference model for non-linear total variability embedding

In the proposed algorithm, it is assumed that the ideal GMM supervector corresponding to a speech utterance $\mathbf{X}$ is obtained through a non-linear mapping of a hidden variable onto the total variability space. Based on this assumption, the ideal GMM supervector is generated from a latent variable $\mathbf{z}$ as follows:

$$\mathbf{m}(\mathbf{X}) = \mathbf{u} + \mathbf{g}(\mathbf{z}(\mathbf{X})) \tag{3.4}$$

where $\mathbf{g}$ is a non-linear function which transforms the hidden variable $\mathbf{z}(\mathbf{X})$ to the adaptation factor representing the variability of the ideal GMM supervector $\mathbf{m}(\mathbf{X})$. In order to find the optimal function $\mathbf{g}$ and the hidden variable $\mathbf{z}(\mathbf{X})$, we apply a VAE model consisting of an encoder and a decoder network as shown in Figure 3.1.

In the proposed VAE architecture, the encoder network outputs an estimate of the hidden variable and the decoder network serves as the non-linear mapping function $\mathbf{g}$.

Analogous to the i-vector adaptation framework, the main task of the proposed VAE architecture is to generate a GMM so as to maximize the likelihood given the Baum-Welch statistics of the utterance. The encoder of the proposed system serves as a non-linear variability factor extraction model. Similar to the i-vector extractor, the encoder network takes the 0^{th} and 1^{st} order Baum-Welch statistics of a given utterance $\mathbf{X}$ as input and generates the parametric distribution of the latent variable. The latent variable $\mathbf{z}$ is assumed to be a random variable following Gaussian distribution and each component of $\mathbf{z}$ is assumed to be uncorrelated with each other. In order to infer the distribution of the latent variable $\mathbf{z}(\mathbf{X})$, it is sufficient for the encoder to generate the mean and the variance of $\mathbf{z}(\mathbf{X})$. The decoder of the proposed system acts as the GMM adaptation model, generating the GMM supervector from the given latent variable according to the maximum likelihood criterion.

3.3.1 Maximum likelihood training

Once the GMM supervector $\hat{\mathbf{m}}(\mathbf{X})$ is generated at the output layer of the decoder, the log-likelihood conditioned on the latent variable $\mathbf{z}(\mathbf{X})$ can be defined as

$$\begin{aligned}\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X})) = \sum_{c=1}^{C}(n_c(\mathbf{X})\log\frac{1}{(2\pi)^{F/2}|\mathbf{\Sigma}_c|^{1/2}} \\ -\frac{1}{2}\sum_{l=1}^{L}\gamma_l(c)(\mathbf{x}_l-\hat{\mathbf{m}}_c(\mathbf{X}))^t\mathbf{\Sigma}_c^{-1}(\mathbf{x}_l-\hat{\mathbf{m}}_c(\mathbf{X}))) \qquad (3.5)\end{aligned}$$

where $\hat{\mathbf{m}}_c(\mathbf{X})$ denotes the c^{th} component of $\hat{\mathbf{m}}(\mathbf{X})$. Using the Jensen's inequality, the lower bound of the marginal log-likelihood can be obtained as follows:

$$\begin{aligned}\log P(\mathbf{X}|\phi,\theta) =& \log \mathbb{E}_{\mathbf{z}}[P(\mathbf{X}|\phi,\theta,\mathbf{z})] \\ \geq& \mathbb{E}_{\mathbf{z}}[\log P(\mathbf{X}|\phi,\theta,\mathbf{z})]. \end{aligned} \tag{3.6}$$

The marginal log-likelihood can be indirectly maximized by maximizing the expectation of the conditioned log-likelihood (3.5) with respect to the latent variable $\mathbf{z}$. The reparameterization trick in [27] can be utilized to compute the Monte Carlo estimate of the log-likelihood lower bound as given by

$$\mathbb{E}_{\mathbf{z}}[\log P(\mathbf{X}|\phi,\theta,\mathbf{z})] \simeq \frac{1}{S}\sum_{s=1}^{S} \log P(\mathbf{X}|\phi,\theta,\mathbf{z}_s(\mathbf{X})) \tag{3.7}$$

where S is the number of samples used for estimation and $\mathbf{z}_s(\mathbf{X})$ is the reparameterized latent variable defined as follows:

$$\mathbf{z}_s(\mathbf{X}) = \mu(\mathbf{X}) + \sigma(\mathbf{X})\epsilon_s. \tag{3.8}$$

In (3.8), $\epsilon_s \sim \mathcal{N}(0, \mathbf{I})$ is an auxiliary noise variable, $\mu(\mathbf{X})$ and $\sigma(\mathbf{X})$ are respectively the mean and standard deviation of the latent variable $\mathbf{z}(\mathbf{X})$ generated from the encoder network. By replacing the reconstruction error term of the VAE objective function (3.3) with the estimated log-likelihood lower bound, the objective function of the proposed system can be written as

$$\begin{aligned} E_{Prop}(\mathbf{X}) =& D_{KL}(q_\phi(\mathbf{z}|\mathbf{X}) \| p_\theta(\mathbf{z})) \\ & - \frac{1}{S}\sum_{s=1}^{S} \log P(\mathbf{X}|\phi,\theta,\mathbf{z}_s(\mathbf{X})). \end{aligned} \tag{3.9}$$

From (3.9), it is seen that the proposed VAE is trained not only to maximize the similarity between the prior and posterior distributions of the latent variable, but

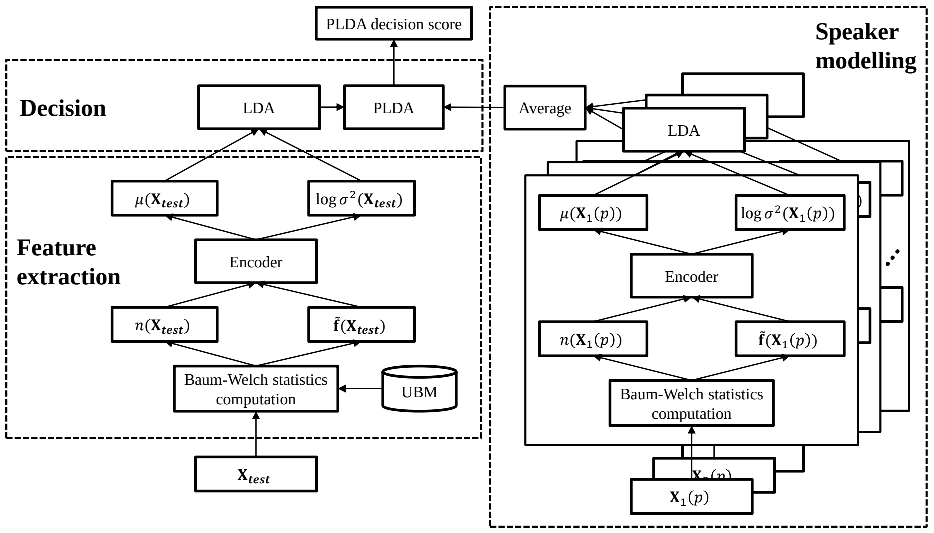


Figure 3.2: Flow chart of the speaker verification process using the proposed feature extraction scheme.

also to maximize the log-likelihood of the generated GMM by minimizing E_{Prop} via error back-propagation. Moreover, we assume that the prior distribution for $\mathbf{z}$ to be $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \mathbf{I})$ analogous to the prior for $\mathbf{w}$ in the i-vector framework.

3.3.2 Non-linear feature extraction and speaker verification

The encoder network of the proposed VAE generates the latent variable mean $\mu(\mathbf{X})$ and the log-variance $\log\sigma^2(\mathbf{X})$. Once the VAE has been trained, the encoder network is used as a feature extraction model as shown in Figure 3.2. Similar to the conventional i-vector extractor, the encoder network takes the Baum-Welch statistics of the input speech utterance and generates a random variable with a Gaussian distribution, which contains essential information for modeling an utterance-dependent GMM. The mean of the latent variable $\mu(\mathbf{X})$ is exploited as a compact representation of the speech's distributive pattern. Moreover, the variance of the latent variable

$\sigma^2(\mathbf{X})$ is used as a proxy for the uncertainty caused by the short duration of the given speech samples. The features extracted by the proposed VAE can be transformed via feature compensation techniques (e.g., linear discriminant analysis, LDA) in order to improve the discriminability of the features.

Given a set of $N(p)$ enrollment speech samples spoken by an arbitrary speaker p

$$\mathbf{X}(p) = \{\mathbf{X}_1(p), \mathbf{X}_2(p), \cdots, \mathbf{X}_{N(p)}(p)\}, \tag{3.10}$$

the speaker model for p is obtained by averaging the features extracted from each speech sample. To determine whether a test utterance $\mathbf{X}_{test}$ is spoken by the speaker p, analogous to the i-vector framework, PLDA is used to compute the similarity between the feature extracted from $\mathbf{X}_{test}$ and the speaker model of p.

Unlike the conventional i-vector framework, which only uses the mean of the latent variable as feature, the proposed scheme utilizes both the mean and variance of the latent variable to take the uncertainty into account. Providing the speaker decision model (e.g., PLDA) with information about the uncertainty within the input speech, which is represented by the variance of the latent variable, may improve the speaker recognition performance.

3.4 Experiments

3.4.1 Databases

In order to evaluate the performance of the proposed technique in random digits speaker verification task, a set of experiments were conducted using the TIDIGITS dataset. The TIDIGITS dataset contains 25096 clean utterances spoken by 111 male, 114 female, 50 boy and 51 girl speakers [30] For each of the 326 speakers in the

TIDIGITS dataset, a set of isolated digits and 2-7 digit sequences were spoken. The TIDIGITS dataset was split into two subsets, each containing 12548 utterances from all the 326 speakers, and they were separately used as the enrollment and trial data. In the TIDIGITS experiments, TIMIT dataset [31] was used for training the UBM, total variability matrix, and the embedding networks.

3.4.2 Experimental setup

The acoustic features used in the experiments were 19 dimensional Mel-frequency cepstral coefficients (MFCCs) and the log-energy extracted at every 10 ms, using a 20 ms Hamming window via the SPro library [32]. Together with the delta and delta-delta of the 19 dimensional MFCCs and the log-energy, the frame-level acoustic feature used in our experiments was given by a 60 dimensional vector.

We trained the UBM containing 32 mixture components in a gender- and age-independent manner, using all the speech utterances in the TIMIT dataset. Training the UBM, total variability matrix, and the i-vector extraction were done by using the MSR Identity Toolbox via MATLAB [33]. The encoder and decoder of the VAEs were configured to have a single hidden layer with 4096 ReLU nodes, and the dimensionality of the latent variables were set to be 200. The implementation of the VAEs was done using Tensorflow [34] and trained using the AdaGrad optimization technique [35]. Also, dropout [36] with a fraction of 0.8 and L2 regularization with a weight of 0.01 were applied for training all the VAEs, and the Baum-Welch statistics extracted from the entire TIMIT dataset were used as training data. A total of 100 samples were used for reparameterization shown in (15).

For all the extracted utterance-level features, linear discriminant analysis (LDA) [15] was applied for feature compensation and the dimensionality was finally reduced to 200. PLDA [37] was used for speaker verification, and the speaker subspace dimension was set to be 200.

Four performance measures were evaluated in our experiments: classification error (Class. err.), EER, minimum NIST SRE 2008 DCF (DCF08), minimum NIST SRE 2010 DCF (DCF10). The classification error was measured while performing a speaker identification task where each trial utterance was compared with all the enrolled speakers via PLDA, and the enrolled speaker with the highest score was chosen as the identified speaker. Then the ratio of the number of wrongly classified trial samples to the total number of trial samples represents the classification error. The EER and minimum DCFs are widely used measures for speaker verification where the EER indicates the error when the false alarm rate (FAR) and the false reject rate (FRR) are the same [15], and the minimum DCFs represent the decision cost obtained with different weights to FAR and FRR. The NIST 2008 DCF [38] gives the same penalty for both FAR and FRR, whereas the NIST 2010 DCF [39] gives more penalty to FRR.

3.4.3 Effect of the duration on the latent variable

In order to investigate the capability of the latent variable for capturing the uncertainty caused by short duration, the differential entropies of the latent variables were computed. The differential entropy, or the continuous random variable entropy, measures the average uncertainty of a random variable. Since the latent variable $\mathbf{z}(\mathbf{X})$ in the proposed VAE has a Gaussian distribution, the differential entropy can be formulated as follows:

$$h(\mathbf{z}(\mathbf{X})) = \frac{1}{2}\log(2\pi e)^K + \frac{1}{2}\log\prod_{k=1}^{K}\sigma_k^2(\mathbf{X}). \tag{3.11}$$

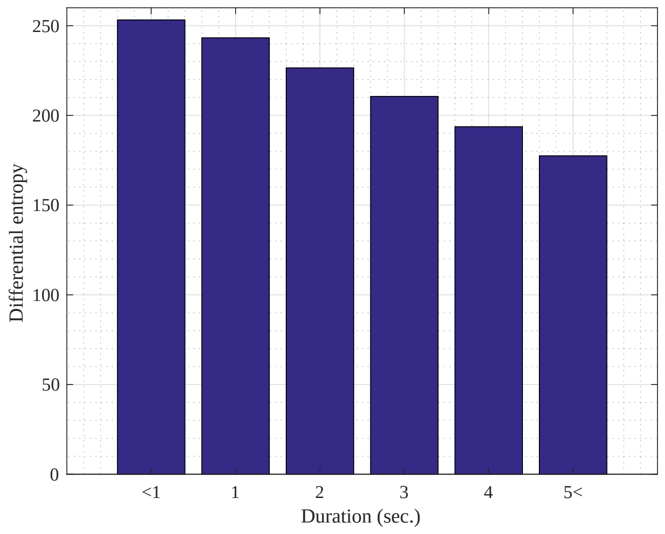


Figure 3.3: Average differential entropy computed using the latent variable variance extracted from the proposed VAE on different durations.

In (3.11), K represents the dimensionality of the latent variable and $\sigma_k^2(\mathbf{X})$ is the k^{th} element of $\sigma^2(\mathbf{X})$. From each speech sample in the entire TIDIGITS dataset, 200 dimensional latent variable variance was generated using the encoder network of the proposed framework and used for computing the differential entropy.

In Figure 3.3, the differential entropies averaged in 6 different duration groups (i.e. less than 1 second, 1-2 seconds, 2-3 seconds, 3-4 seconds, 4-5 seconds, and more than 5 seconds) are shown. As shown in the result, the differential entropy computed using the variance of the latent variable gradually decreases as the duration increases. Despite a rather small time difference between the 1^{st} duration group (i.e. less than 1 second) and the 6^{th} duration group (i.e. more than 5 seconds), the relative decrement in entropy was 29.91%. This proves that the latent variable variance extracted from the proposed system is capable of indicating the uncertainty caused by the short duration.

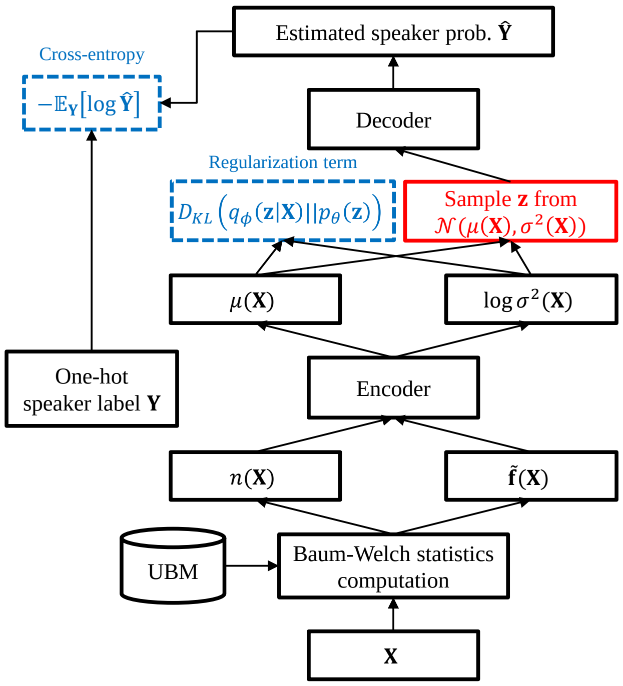

Estimated speaker prob. $\hat{\mathbf{Y}}$
Cross-entropy
$-\mathbb{E}_{\mathbf{Y}}[\log \hat{\mathbf{Y}}]$
Decoder
Regularization term
$D_{KL}\left(q_\phi(\mathbf{z}|\mathbf{X})||p_\theta(\mathbf{z})\right)$
Sample $\mathbf{z}$ from $\mathcal{N}(\mu(\mathbf{X}), \sigma^2(\mathbf{X}))$
$\mu(\mathbf{X})$
$\log \sigma^2(\mathbf{X})$
One-hot speaker label $\mathbf{Y}$
Encoder
$n(\mathbf{X})$
$\tilde{\mathbf{f}}(\mathbf{X})$
UBM
Baum-Welch statistics computation
$\mathbf{X}$

Figure 3.4: Network structure of the baseline VAE *Classify*.

3.4.4 Experiments with VAEs

To verify the performance of the proposed VAE trained with the log-likelihood-based reconstruction error function, we conducted a series of speaker recognition experiments over the TIDIGITS dataset. For performance comparison, we also applied various feature extraction approaches. The approaches which were compared with each other in these experiments are as follows:

- *i-vector*: standard 200 dimensional i-vector,
- *Autoencode*: VAE trained to minimize the cross-entropy between the input Baum-Welch statistics and the reconstructed output Baum-Welch statistics,

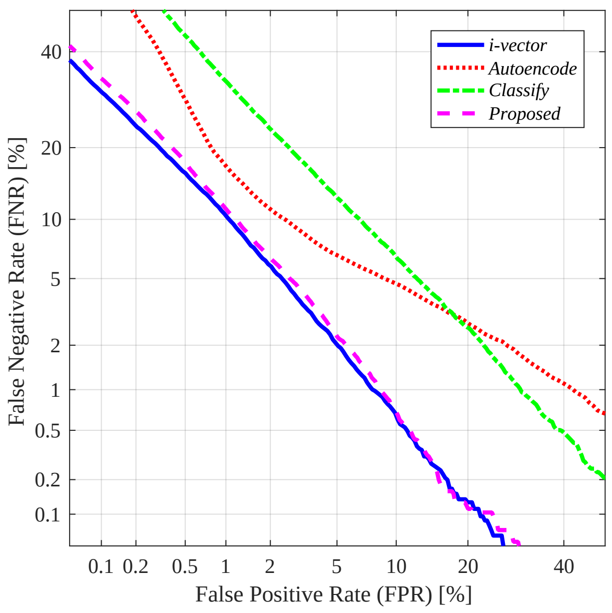


Figure 3.5: DET curves of the speaker verification experiments using the i-vector and the mean latent variables extracted from VAEs trained for different tasks.

- *Classify*: VAE trained to minimize the cross-entropy between the softmax output and the one-hot speaker label,
- *Proposed*: the proposed VAE trained to minimize the negative log-likelihood-based reconstruction error.

Autoencode is a standard VAE for reconstructing the input at the output, which was trained to minimize E_{VAE} (3.3) given the Baum-Welch statistics as input. On the other hand, *Classify* is a VAE for estimating the speaker label, which was trained to minimize the following loss function:

$$E_{Class}(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})) - \mathbb{E}_{\mathbf{Y}}[\log \hat{\mathbf{Y}}] \tag{3.12}$$

where $\mathbf{Y}$ denotes the one-hot speaker label of utterance $\mathbf{X}$ and $\hat{\mathbf{Y}}$ is the softmax

Table 3.1: Comparison of results between using the i-vector and the mean latent variables extracted from VAEs trained for different tasks.

	Class. err. [%]	EER [%]	DCF08	DCF10
i-vector	12.62	**3.36**	**2.00**	**0.07**
Autoencode	24.59	6.06	2.69	0.09
Classify	40.01	8.13	4.21	0.10
Proposed	**11.89**	3.61	2.09	0.07

output of the decoder network. The network structure for *Classify* is depicted in Figure 3.4. In this experiment, only the mean vectors of the latent variables were used for *Autoencode*, *Classify*, and *Proposed*.

The results shown in Table 3.1 tell us that the VAEs trained with the conventional criteria (i.e. *Autoencode* and *Classify*) perform poorly compared to the standard i-vector. On the other hand, the proposed VAE with likelihood-based reconstruction error was shown to provide better performance for speaker recognition than the other methods. The feature extracted using the VAE trained with the proposed criterion provided comparable verification performance (i.e. EER, DCF08, DCF10) to the conventional i-vector feature. Moreover, in terms of classification, the proposed framework outperformed the i-vector framework with a relative improvement of 5.8% in classification error. Figure 3.5 shows the DET curves obtained from the four tested approaches.

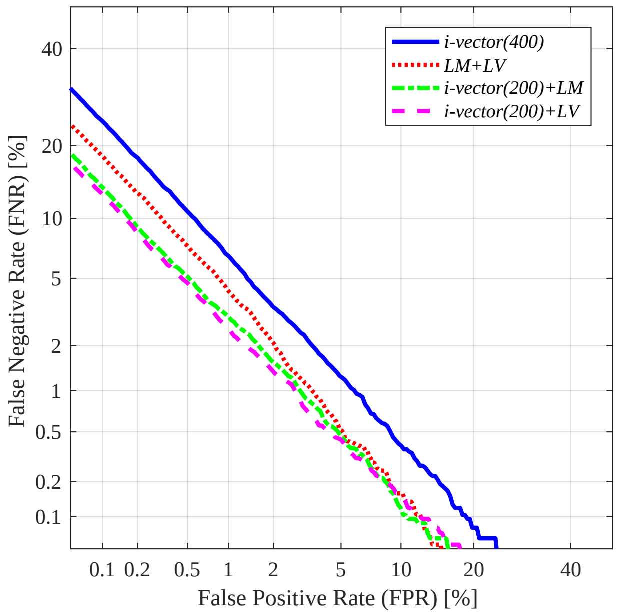


Figure 3.6: DET curves of the speaker verification experiments using 400 dimensional i-vector and combinations of two features out of 200 dimensional i-vector, latent variable mean, and the log-variance of the latent variable.

3.4.5 Feature-level fusion of i-vector and latent variable

In this subsection, we tested the features obtained by augmenting the conventional i-vector with the mean and variance of the latent variable extracted from the proposed VAE. For performance comparison, we applied the following six different feature sets:

- *i-vector(400)*: standard 400 dimensional i-vector,
- *i-vector(600)*: standard 600 dimensional i-vector,
- *LM+LV*: concatenation of the 200 dimensional latent variable mean and the log-variance, resulting in a 400 dimensional vector,

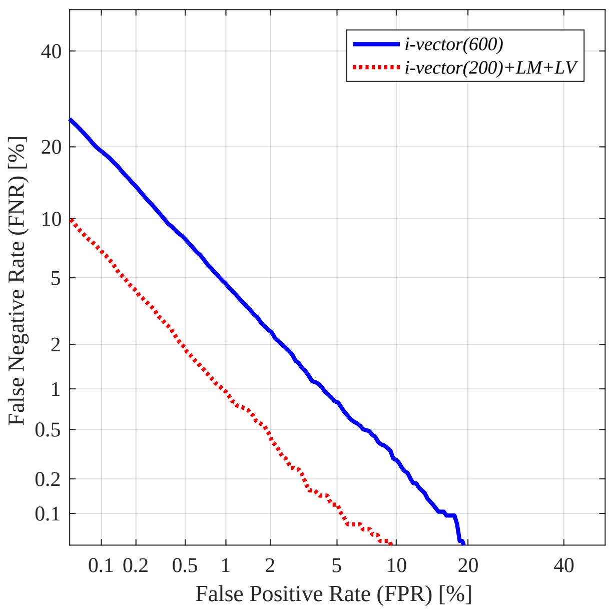


Figure 3.7: DET curves of the speaker verification experiments using 600 dimensional i-vector and combined feature of 200 dimensional i-vector, latent variable mean, and the log-variance of the latent variable.

- *i-vector(200)+LM*: concatenation of the 200 dimensional i-vector and the 200 dimensional latent variable mean, resulting in a 400 dimensional vector,
- *i-vector(200)+LV*: concatenation of the 200 dimensional i-vector and the 200 dimensional latent variable log-variance, resulting in a 400 dimensional vector,
- *i-vector(200)+LM+LV*: concatenation of the 200 dimensional i-vector and the 200 dimensional latent variable mean and log-variance, resulting in a 600 dimensional vector.

As seen from Table 3.2 and Figure 3.6, the augmentation of the latent variable greatly improved the performance in all the tested cases. By using only the mean and log-variance of the latent variable together (i.e. *LM+LV*), a relative improvement of

Table 3.2: Comparison of results between various feature-level fusions of the conventional i-vector, mean and log-variance of the latent variable extracted from the proposed VAE.

	Class. err. [%]	EER [%]	DCF08	DCF10
i-vector(400)	7.67	2.68	1.54	0.06
LM+LV	6.94	2.03	1.23	0.05
i-vector(200)+LM	5.36	1.78	0.97	0.05
i-vector(200)+LV	**4.99**	**1.65**	**0.94**	**0.04**
i-vector(600)	5.07	2.17	1.29	0.05
i-vector(200)+LM+LV	**2.75**	**0.97**	**0.61**	**0.03**

24.25% was achieved in terms of EER, compared to the conventional i-vector with the same dimension (i.e. *i-vector(400)*). The concatenation of the standard i-vector and the latent variable mean (i.e. *i-vector(200)+LM*) also improved the performance. Especially in terms of EER, *i-vector(200)+LM* achieved a relative improvement of 33.58% compared to *i-vector(400)*. This improvement may be attributed to the non-linear feature extraction process. Since the latent variable mean is trained to encode the various variability within the distributive pattern of the given utterance via a non-linear process, it may contain information not obtainable from the linearly extracted i-vector. Thus by supplementing the information ignored by the i-vector extraction process, a better representation of the speech can be obtained.

The best verification and identification performance out of all the 400 dimensional features (i.e. *i-vector(400)*, *LM+LV*, *i-vector(200)+LM*, and *i-vector(200)+LV*) was obtained when concatenating the standard i-vector and the latent variable log-variance (i.e. *i-vector(200)+LV*). *i-vector(200)+LV* achieved a relative improvement of 38.43% in EER and 34.94% in classification error compared to *i-vector(400).* This may be due to the capability of the latent variable variance in capturing the amount of uncertainty, which allows the decision score to take advantage of the duration dependent reliability.

Concatenating the standard i-vector with both the mean and log-variance of the latent variable (i.e. *i-vector(200)+LM+LV*) further improved the speaker recognition performance. Using the *i-vector(200)+LM+LV* achieved a relative improvement of 55.30% in terms of EER, compared to the standard i-vector with the same dimension (i.e. *i-vector(600)*). Figure 3.7 shows the DET curves obtained when *i-vector(200)+LM+LV* and *i-vector(600)* were applied.

3.4.6 Score-level fusion of i-vector and latent variable

In this subsection, we present the experimental results obtained from a speaker recognition task where the decision is made by fusing the PLDA scores of i-vector features and VAE-based features. Given a set of independently computed PLDA scores S_r, $r = 1, \cdots, R$, the fused score S_{fused} was computed by simply adding them as

$$S_{fused} = \sum_{r=1}^{R} S_r. \tag{3.13}$$

We compared the following scoring schemes:

- *i-vector*: PLDA score obtained by using the standard 200 dimensional i-vector,
- *LM*: PLDA score obtained by using the 200 dimensional latent variable mean,
- *LV*: PLDA score obtained by using the 200 dimensional latent variable log-variance,

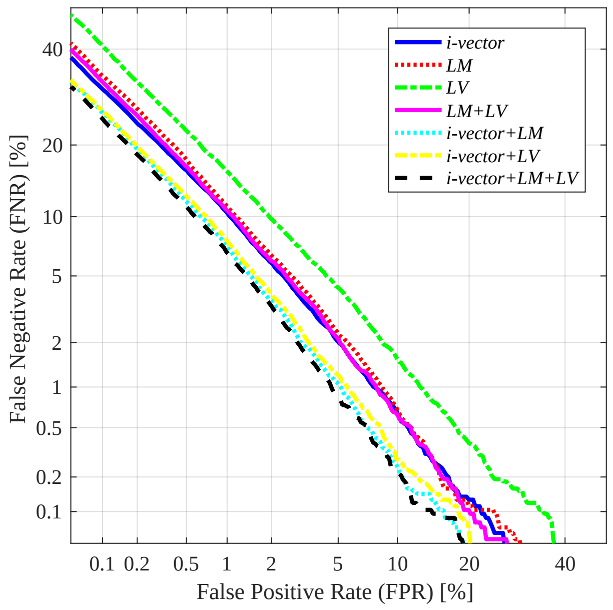


Figure 3.8: DET curves of the speaker verification experiments using various score-level fusions of the conventional i-vector, mean and log-variance of the latent variable extracted from the proposed VAE.

- *i-vector+LM*: fusion of the PLDA scores obtained by using the 200 dimensional i-vector and the 200 dimensional latent variable mean,
- *i-vector+LV*: fusion of the PLDA scores obtained by using the 200 dimensional i-vector and the 200 dimensional latent variable log-variance,
- *LM+LV*: fusion of the PLDA scores obtained by using the latent variable mean and log-variance,
- *i-vector+LM+LV*: fusion of the PLDA scores obtained by using the standard 200 dimensional i-vector, 200 dimensional latent variable mean and log-variance.

Table 3.3: Comparison of results between various score-level fusions of the conventional i-vector, mean and log-variance of the latent variable extracted from the proposed VAE.

	Class. err. [%]	EER [%]	DCF08	DCF10
i-vector	12.62	3.36	2.00	0.07
LM	11.89	3.61	2.09	0.07
LV	17.78	4.65	2.57	0.08
i-vector+LM	7.03	2.63	1.63	**0.06**
i-vector+LV	7.39	2.76	1.69	**0.06**
LM+LV	10.26	3.50	2.02	0.07
i-vector+LM+LV	**5.75**	**2.49**	**1.57**	**0.06**

Table 3.3 and Figure 3.8 give the results obtained through these scoring schemes. As shown in the results, using the latent variable mean and log-variance vectors as standalone features yielded comparable performance to the conventional i-vector method (i.e. *LM* and *LV*). Also, fusing the latent variable-based scores with the score provided by the standard i-vector feature further improved the performance (i.e. *i-vector+LM* and *i-vector+LV*). The best score-level fusion performance was obtained by fusing all the scores obtained by the standard i-vector, the latent variable mean and log-variance vector (i.e. *i-vector+LM+LV*), achieving a relative improvement of 25.89% in terms of EER compared to *i-vector*. However, the performance improvement produced by the score-level fusion methods was relatively smaller than the feature-level fusion methods presented in Table 3.2. This may be due to the fact that the score-level fusion methods compute the scores of the i-vector and the la-

tent variable-based features independently, and as a result the final score cannot be considered an optimal way to utilize their joint information.

3.5 Summary

In this Chapter, a novel deep learning model-based utterance-level feature extractor for speaker recognition was proposed. In order to capture the variability that has not been fully represented by the linear projection in the traditional i-vector framework, we designed a VAE for GMM adaptation and exploited the latent variable as the non-linear representation of the variability in the given speech. Analogous to the standard VAE, the proposed architecture is composed of an encoder and a decoder network where the former estimates the distribution of the latent variable given the Baum-Welch statistics of the speech and the latter generates the ideal GMM supervector from the latent variable. Moreover, to take the uncertainty caused by short duration speech utterances into account while extracting the feature, the VAE is trained to generate a GMM supervector in a way to maximize the likelihood. The training stage of the proposed VAE uses a likelihood-based error function instead of the conventional reconstruction errors (e.g., cross-entropy).

To investigate the characteristics of the features extracted from the proposed system in a practical scenario, we conducted a set of random digits sequence experiments using the TIDIGITS dataset. We observed that the variance of the latent variable generated from the proposed network apparently demonstrates the level of uncertainty which gradually decreases as the duration of the speech increases. Also, using the mean and variance of the latent variable as feature provided comparable performance to the conventional i-vector and further improved when used in conjunction with the i-vector. The best performance was achieved by feature-level fusion of the i-vector, mean and variance of the latent variable.

Chapter 4

Adversarially learned total variability embedding for speaker recognition with random digits strings

4.1 Introduction

In Chapter 3, we successfully exploited the variational autoencoder (VAE) framework [27], [28] to extract an utterance-level feature to capture the total variability and the uncertainty of the speech in a non-linear fashion. The VAE has an autoencoder-like structure and assumes that the input data is generated through a directed graphical model induced by a latent variable. The latent variable of the VAE-based feature extraction model in [12] serves as an utterance-level representation and has shown better performance than the conventional i-vector in coping with the uncertainty caused by the short duration. Moreover, we could observe that the variance of the latent variable can be used as a proxy for the uncertainty caused by the short duration of the given speech samples. However, as mentioned in [40] and [41], since the Kullback-Leibler divergence (KL divergence) regularization term in the VAE objective function encourages the variational posterior to be closer to the prior, which leads to less informative latent variable representations. Due to this limitation, the embedding extracted from the VAE-based feature extractor may lack crucial speaker-dependent information, resulting in limited performance of the overall speaker recognition system.

In order to overcome this problem, we propose a novel approach to train a deep learning-based embedding network using the adversarial learning framework. The proposed method adopts an adversarially learned inference (ALI) model [42] to non-linearly express the total variability and uncertainty of the given speech. Analogous to the VAE-based feature extractor in [12], the proposed model is trained according to the maximum likelihood criterion and the latent variable serves as the utterance-level feature representation. However instead of simply regularizing the latent variable via KL divergence, the proposed method uses a discriminator network to make sure that the generated latent variable and GMM supervector are close to the latent prior distribution and the GMM obtained through maximum a posteriori (MAP) adaptation, respectively. While training the proposed network, the parameters of the feature extraction and the discriminator networks are updated competitively; the feature extraction network tries to generate a more realistic GMM supervector and latent variable while the discriminator network focuses on distinguishing the generated GMM and latent variable from the real ones. By training the feature extraction network in this adversarial fashion, the proposed system is expected to provide utterance-level features which can capture more prominent speaker relevant characteristics and the uncertainty within the given speech utterance. Similarly to the VAE-based feature extractor, the utterance-level features extracted from the proposed model can substitute the conventional i-vector extraction module without any difficulty.

To evaluate the performance of the proposed method, along with the VAE-based feature extraction scheme and the conventional i-vector framework, we conducted a set of speaker verification experiments using the TIDIGITS dataset. Experimental results show that the proposed method outperforms the standard i-vector framework and the VAE-based method in terms of equal error rate (EER). It is also interesting to see that impressive performance improvement is observed when the features extracted from the proposed method and the conventional i-vector are augmented together. This implies that the feature extracted using the proposed method is complementary to that extracted from the conventional i-vector framework.

4.2 Adversarially learned inference

Alike other variants of the generative adversarial network (GAN) [43], ALI aims to train a network for generating a realistic data sample with the help of a discriminator network, which tries to predict whether the input data is real or generated [42]. However, unlike the ordinary GAN framework which cannot analyze the data at an abstract level, ALI integrates an inference network for estimating the random latent variable $\mathbf{z}$ from the input data. As depicted in Fig. 4.1, ALI is composed of three directed networks: encoder, decoder, and joint discriminator networks. Analogous

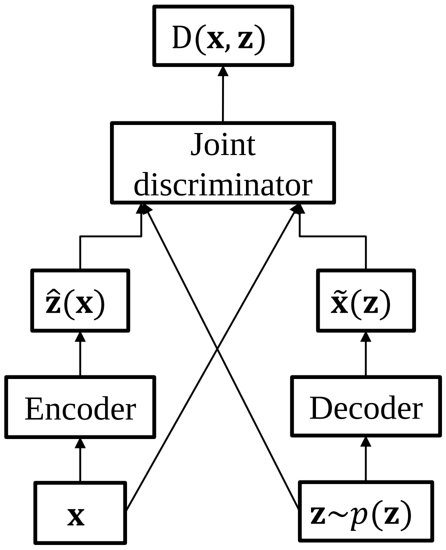


Figure 4.1: Flow chart of the ALI framework.

to the VAE network, the encoder network outputs the mean and variance of the posterior distribution $p(\mathbf{z}|\mathbf{x})$ given an observation $\mathbf{x}$. On the other hand, the decoder generates the data sample from a latent variable sampled from a prior distribution $p(\mathbf{z})$.

The encoder and decoder networks represent the joint probability distributions of the latent variable $\mathbf{z}$ and the observed data $\mathbf{x}$ as follows:

$$\begin{aligned} q_\phi(\mathbf{x},\mathbf{z}) &= q_\phi(\mathbf{x})q_\phi(\mathbf{z}|\mathbf{x}), \\ p_\theta(\mathbf{x},\mathbf{z}) &= p_\theta(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z}). \end{aligned} \tag{4.1}$$

In (4.1), ϕ, θ, $q_\phi(\mathbf{x},\mathbf{z})$ and $p_\theta(\mathbf{x},\mathbf{z})$ denote the encoder parameters, decoder parameters, and the joint distributions represented by the encoder and decoder, respectively. The encoder marginal probability $q_\phi(\mathbf{x})$ represents the real data distribution and the decoder marginal probability $p_\theta(\mathbf{z})$ is the prior distribution of the latent variable, usually specified as a standard normal distribution. The conditional probability distributions $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ represent the inferred latent distribution output by the encoder and the distribution of the data generated by the decoder, respectively.

The joint discriminator takes a joint pair of the data and the latent variable as input and distinguishes between the samples from the encoder $(\mathbf{x}, \hat{\mathbf{z}}(\mathbf{x}))\sim q_\phi(\mathbf{x}, \mathbf{z})$ and the ones from the decoder $(\tilde{\mathbf{x}}(\mathbf{z}), \mathbf{z})\sim p_\theta(\mathbf{x}, \mathbf{z})$. The discriminator output $D(\mathbf{x}, \mathbf{z})$ is sigmoidal, ideally having a value close to 0 if the samples are drawn from $p_\theta(\mathbf{x}, \mathbf{z})$ and 1 if drawn from $q_\phi(\mathbf{x}, \mathbf{z})$.

The encoder, decoder and the joint discriminator networks of the ALI are trained adversarially by minimizing the following objective functions:

$$\begin{aligned} E_D = &- \mathbb{E}_{(\mathbf{x},\mathbf{z})\sim q_\phi(\mathbf{x},\mathbf{z})}[\log(D(\mathbf{x},\mathbf{z}))] \\ &- \mathbb{E}_{(\mathbf{x},\mathbf{z})\sim p_\theta(\mathbf{x},\mathbf{z})}[\log(1 - D(\mathbf{x},\mathbf{z}))], \end{aligned} \tag{4.2}$$

$$\begin{aligned} E_G = &- \mathbb{E}_{(\mathbf{x},\mathbf{z})\sim q_\phi(\mathbf{x},\mathbf{z})}[\log(1 - D(\mathbf{x},\mathbf{z}))] \\ &- \mathbb{E}_{(\mathbf{x},\mathbf{z})\sim p_\theta(\mathbf{x},\mathbf{z})}[\log(D(\mathbf{x},\mathbf{z}))]. \end{aligned} \tag{4.3}$$

In (4.2) and (4.3), E_D denotes the discriminator loss function and E_G is the generator loss function. The joint discriminator network is trained to minimize E_D, which decreases as the network distinguishes between the samples from the encoder and the decoder better and the parameters θ, ϕ of the generator (i.e. encoder and decoder networks) are updated to minimize E_G.

4.3 Adversarially learned feature extraction

The proposed algorithm assumes that the variability of the utterance-dependent GMM supervector of the UBM can be represented by a non-linear projection of a

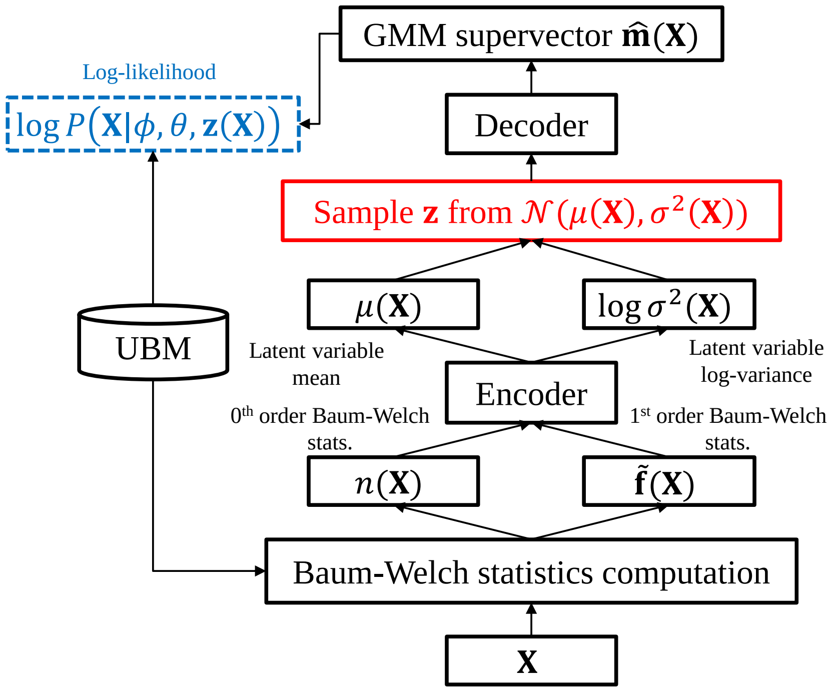

GMM supervector $\hat{\mathbf{m}}(\mathbf{X})$
Log-likelihood
$\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))$
Decoder
Sample $\mathbf{z}$ from $\mathcal{N}(\mu(\mathbf{X}),\sigma^2(\mathbf{X}))$
$\mu(\mathbf{X})$
$\log\sigma^2(\mathbf{X})$
UBM
Latent variable mean
Latent variable log-variance
Encoder
0th order Baum-Welch stats.
1st order Baum-Welch stats.
$n(\mathbf{X})$
$\tilde{\mathbf{f}}(\mathbf{X})$
Baum-Welch statistics computation
$\mathbf{X}$

Figure 4.2: Maximum likelihood training scheme for the proposed non-linear feature extractor. Red shows the sampling operation. Blue shows the loss term.

hidden variable onto the variability space as follows:

$$\mathbf{m}(\mathbf{X}) = \mathbf{u} + \mathbf{g}(\mathbf{z}(\mathbf{X})) \tag{4.4}$$

where $\mathbf{g}$ is a non-linear function which transforms the hidden variable $\mathbf{z}(\mathbf{X})$ to the total variability of the GMM supervector $\mathbf{m}(\mathbf{X})$. As shown in Fig. 4.2, the proposed scheme employs an encoder network for inferring the hidden variable $\mathbf{z}(\mathbf{X})$ from the observed speech $\mathbf{X}$ and a decoder network for non-linearly mapping $\mathbf{z}(\mathbf{X})$ onto the total variability space to generate an ideal utterance-dependent GMM supervector.

As with the i-vector adaptation framework, the main task of the proposed approach is to generate an utterance-dependent GMM that maximizes the likelihood given the Baum-Welch statistics of the utterance. The encoder network takes the zeroth- and first-order Baum-Welch statistics of the input utterance $\mathbf{X}$ and estimates the posterior distribution, namely the mean μ and the variance σ^2 of the latent variable through variational inference. The latent variable $\mathbf{z}$ is a random variable assumed to follow a Gaussian distribution and its components are uncorrelated with each other. The decoder network generates the GMM supervector given the latent variable according to the maximum likelihood criterion.

4.3.1 Maximum likelihood criterion

The log-likelihood of the GMM supervector $\hat{\mathbf{m}}(\mathbf{X})$ generated from the decoder network conditioned on the latent variable $\mathbf{z}(\mathbf{X})$, which is sampled from the posterior distribution $q_\phi(\mathbf{z}|\mathbf{X})$ generated by the encoder network given input observation $\mathbf{X}$, can be written as

$$\begin{aligned}\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X})) = &\sum_{c=1}^{C}(n_c(\mathbf{X})\log\frac{1}{(2\pi)^{F/2}|\mathbf{\Sigma}_c|^{1/2}}\\ &-\frac{1}{2}\sum_{l=1}^{L}\gamma_l(c)(\mathbf{x}_l-\hat{\mathbf{m}}_c(\mathbf{X}))^t\mathbf{\Sigma}_c^{-1}(\mathbf{x}_l-\hat{\mathbf{m}}_c(\mathbf{X})))\end{aligned} \tag{4.5}$$

where $\hat{\mathbf{m}}_c(\mathbf{X})$ denotes the c^{th} component of $\hat{\mathbf{m}}(\mathbf{X})$. Instead of directly maximizing the marginal log-likelihood $\log P(\mathbf{X}|\phi,\theta) = \log\mathbb{E}_{\mathbf{z}(\mathbf{X})\sim q_\phi(\mathbf{z}|\mathbf{X})}[P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))]$, which is practically intractable, the proposed algorithm maximizes its lower bound which can be obtained via Jensen's inequality as follows:

$$\begin{aligned}\log P(\mathbf{X}|\phi,\theta) =& \log\mathbb{E}_{\mathbf{z}(\mathbf{X})\sim q_\phi(\mathbf{z}|\mathbf{X})}[P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))]\\ \geq& \mathbb{E}_{\mathbf{z}(\mathbf{X})\sim q_\phi(\mathbf{z}|\mathbf{X})}[\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))].\end{aligned} \tag{4.6}$$

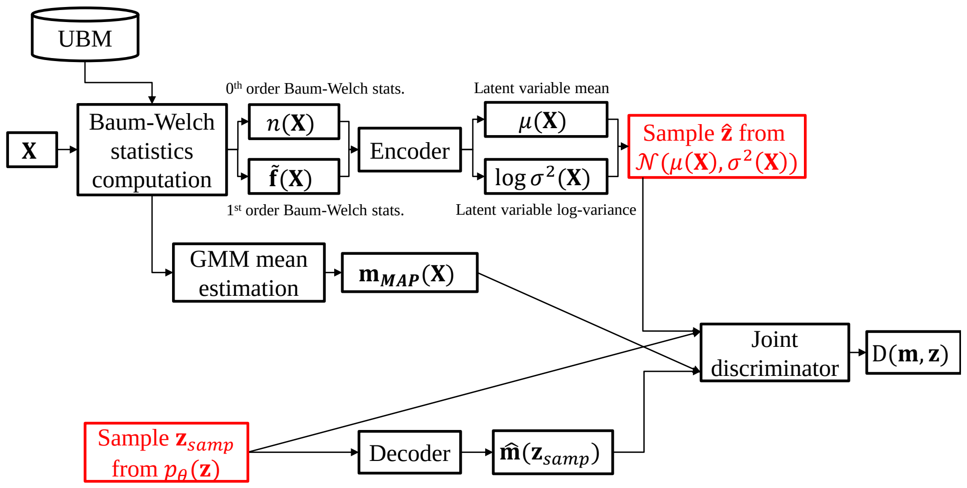

UBM
X
Baum-Welch statistics computation
0th order Baum-Welch stats.
$n(\mathbf{X})$
$\tilde{\mathbf{f}}(\mathbf{X})$
1st order Baum-Welch stats.
Encoder
Latent variable mean
$\mu(\mathbf{X})$
$\log\sigma^2(\mathbf{X})$
Latent variable log-variance
Sample $\hat{\mathbf{z}}$ from $\mathcal{N}(\mu(\mathbf{X}), \sigma^2(\mathbf{X}))$
GMM mean estimation
$\mathbf{m}_{MAP}(\mathbf{X})$
Joint discriminator
$D(\mathbf{m}, \mathbf{z})$
Sample $\mathbf{z}_{samp}$ from $p_\theta(\mathbf{z})$
Decoder
$\hat{\mathbf{m}}(\mathbf{z}_{samp})$

Figure 4.3: ALI-based training scheme for the proposed non-linear feature extractor. Red shows the sampling operations.

Using the reparameterization trick [27], the Monte Carlo estimate of the marginal log-likelihood lower bound can be computed as

$$\begin{aligned}&\mathbb{E}_{\mathbf{z}(\mathbf{X})\sim q_\phi(\mathbf{z}|\mathbf{X})}[\log P(\mathbf{X}|\phi,\theta,\mathbf{z}(\mathbf{X}))]\\ &\qquad\simeq \frac{1}{S}\sum_{s=1}^{S}\log P(\mathbf{X}|\phi,\theta,\mathbf{z}_s(\mathbf{X})),\end{aligned} \tag{4.7}$$

where S is the number of samples used for the estimation and $\mathbf{z}_s(\mathbf{X})$ is the reparameterized latent variable defined by

$$\mathbf{z}_s(\mathbf{X}) = \mu(\mathbf{X}) + \sigma(\mathbf{X})\epsilon_s. \tag{4.8}$$

In (4.8), $\mu(\mathbf{X})$ and $\sigma(\mathbf{X})$ are respectively the mean and standard deviation of the latent variable $\mathbf{z}(\mathbf{X})$ generated from the encoder network, and $\epsilon_s \sim \mathcal{N}(0, \mathbf{I})$ is an auxiliary noise variable.

4.3.2 Adversarially learned inference for non-linear i-vector extraction

In order to ensure that the generated latent variable $\mathbf{z}$ matches its prior distribution and the GMM supervector $\hat{\mathbf{m}}$ well preserves the distributive structure of the GMM driven from the UBM, the proposed scheme utilizes a joint discriminator for regularizing the encoder and decoder parameters. As shown in Fig. 4.3, the joint discriminator of the proposed algorithm takes the GMM supervector and the latent variable and tries to determine whether the input pairs are generated from the encoder or the decoder networks.

However, since the decoder output $\hat{\mathbf{m}}(\mathbf{z})$ does not match the encoder inputs $n(\mathbf{X})$ and $\tilde{f}(\mathbf{X})$, the joint discriminator cannot be applied directly. To alleviate this difficulty, the proposed scheme first estimates the GMM mean vectors via maximum a posteriori (MAP) adaptation [21] given as follows:

$$\mathbf{m}_{c,MAP}(\mathbf{X}) = \frac{\sum_{l=1}^{L} \gamma_l(c)\mathbf{x}_l}{\sum_{l=1}^{L} \gamma_l(c)} = \frac{\tilde{\mathbf{f}}_c(\mathbf{X})}{n_c(\mathbf{X})} + \mathbf{u}_c, \tag{4.9}$$

where $\mathbf{m}_{c,MAP}(\mathbf{X})$ is the estimated c^{th} Gaussian mixture mean given the input speech utterance $\mathbf{X}$. The mean vectors $\mathbf{m}_{c,MAP}(\mathbf{X})$ for $c = 1, ..., C$ are concatenated to form a GMM supervector $\mathbf{m}_{MAP}(\mathbf{X})$.

The joint discriminator takes the joint pair either from the encoder $(\mathbf{m}_{MAP}(\mathbf{X}), \hat{\mathbf{z}}) \sim q_\phi(\mathbf{m}, \mathbf{z})$ or from the decoder $(\hat{\mathbf{m}}(\mathbf{z}_{samp}), \mathbf{z}_{samp}) \sim p_\theta(\mathbf{m}, \mathbf{z})$ as input, where $\hat{\mathbf{z}}$ and $\mathbf{z}_{samp}$ are the latent variables sampled from $\mathcal{N}(\mu(\mathbf{X}), \log\sigma^2(\mathbf{X}))$ and $p_\theta(\mathbf{z})$, respectively, and $\hat{\mathbf{m}}(\mathbf{z}_{samp})$ is the GMM supervector generated by the decoder given $\mathbf{z}_{samp}$. We assume that the prior distribution for the latent variable $p_\theta(\mathbf{z})$ to be $\mathcal{N}(\mathbf{z}|0, \mathbf{I})$, akin to the prior for $\mathbf{w}$ in the i-vector framework. As in (4.2), the discriminator network parameter is trained to minimize the following objective function:

$$\begin{aligned} E_{Prop,D} =& -\mathbb{E}_{(\mathbf{m},\mathbf{z})\sim q_{\phi}(\mathbf{m},\mathbf{z})}[\log(D(\mathbf{m},\mathbf{z}))] \\ & -\mathbb{E}_{(\mathbf{m},\mathbf{z})\sim p_{\theta}(\mathbf{m},\mathbf{z})}[\log(1-D(\mathbf{m},\mathbf{z}))]. \end{aligned} \tag{4.10}$$

By combining the generator loss function of ALI (4.3) and the marginal log-likelihood lower bound (4.6), the objective function of the encoder and decoder networks of the proposed framework can be written as

$$\begin{aligned} E_{Prop,G} =& -\mathbb{E}_{(\mathbf{m},\mathbf{z})\sim q_{\phi}(\mathbf{m},\mathbf{z})}[\log(1-D(\mathbf{m},\mathbf{z}))] \\ & -\mathbb{E}_{(\mathbf{m},\mathbf{z})\sim p_{\theta}(\mathbf{m},\mathbf{z})}[\log(D(\mathbf{m},\mathbf{z}))] \\ & -\frac{1}{S}\sum_{s=1}^{S}\log P(\mathbf{X}|\phi,\theta,\mathbf{z}_s(\mathbf{X})). \end{aligned} \tag{4.11}$$

From (4.11), it is seen that the encoder and decoder networks are trained not only to generate latent variables and GMM supervectors from $q_{\phi}(\mathbf{m},\mathbf{z})$ or $p_{\theta}(\mathbf{m},\mathbf{z})$ that are identical to each other, but also to maximize the log-likelihood of the generated utterance-dependent GMM by minimizing $E_{Prop,G}$ through error back-propagation [29].

4.3.3 Relationship to the VAE-based feature extractor

The VAE-based feature extraction network [12] focuses on maximizing the log-likelihood of the generated GMM by minimizing the following objective function:

$$\begin{aligned} E_{VAE/FE}(\mathbf{X}) =& D_{KL}(q_{\phi}(\mathbf{z}|\mathbf{X})\|p_{\theta}(\mathbf{z})) \\ & -\frac{1}{S}\sum_{s=1}^{S}\log P(\mathbf{X}|\phi,\theta,\mathbf{z}_s(\mathbf{X})). \end{aligned} \tag{4.12}$$

The first term in the RHS of (4.12) is the KL divergence between the prior and the posterior distribution of the latent variable $\mathbf{z}$, which can be viewed as the regularization term. The regularization term forces the encoder network to generate a latent variable distribution that is compatible with $p_\theta(\mathbf{z})$. However, the KL regularization term stretches the latent space over the entire training set to avoid assigning small probability to any training samples [40], [41]. Due to such problem of the KL regularization term, the VAE tends to generate conservative outputs, which usually lack in variety. Especially in the image processing community, it has been reported that the VAE-based image generators result in blurry image samples [42]. In the same manner, the KL regularization term may lead the VAE-based feature extractor to produce utterance-level features with insufficient idiosyncratic representation for the speaker.

The proposed ALI-based feature extraction framework, in contrast, does not regularize the latent variable distribution with a KL divergence term. Instead, the proposed scheme employs a joint discriminator network, which encourages the encoder and decoder networks to generate realistic latent variables and GMM supervectors. Thus the distinctive information within the latent variables generated by the ALI-based feature extractor is less likely to be tightly constrained by its prior distribution.

4.4 Experiments

4.4.1 Databases

In order to evaluate the performances of the baseline systems and the proposed scheme in a condition similar to real-life application where the speech data for training and enrolling are limited and usually have short durations, we performed experiments using the TIMIT dataset [31] as the development set and TIDIGITS dataset [30] as the enrollment and trial sets. The TIMIT dataset contains 6,300

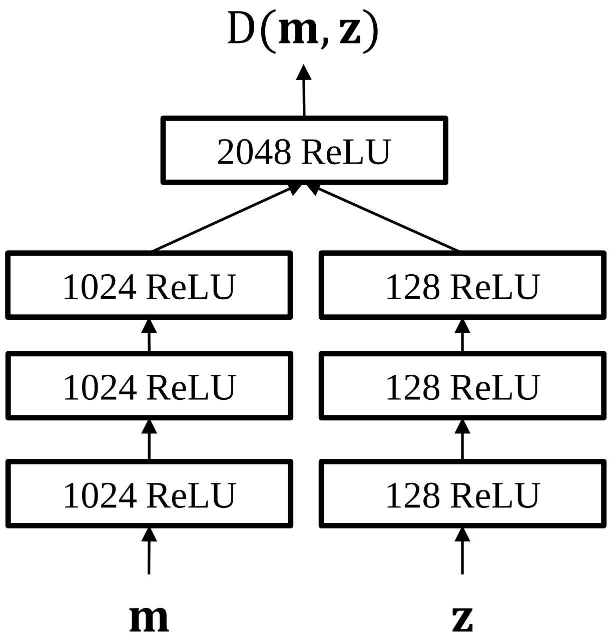


Figure 4.4: Network structure for the discriminator of the ALI-based feature extractor.

clean recorded utterances, 10 utterances spoken by each of 438 male and 192 female speakers. Each utterance in the TIMIT dataset has an average duration of 3 seconds. The TIMIT dataset was used for training the UBM and also used for training the total variability matrix. The TIDIGITS dataset contains 25,096 clean recorded utterances spoken by 111 male, 114 female, 50 boy and 51 girl speakers. Each of the 326 speakers in the TIDIGITS dataset spoke a set of isolated digits and 2-7 digit sequences. The TIDIGITS dataset was split into two subsets, each containing 12,548 utterances from all 326 speakers, and they were separately used as the enrollment and trial data.

4.4.2 Experimental setup

The acoustic features used in the experiments were 19-dimensional Mel-frequency cepstral coefficients (MFCCs) and the log-energy extracted at every 10 ms, using a 20 ms Hamming window via the SPro library [32]. Together with the delta and delta-delta of the 19-dimensional MFCCs and the log-energy, the frame-level acoustic feature used in our experiments was a 60 dimensional vector.

We trained a UBM containing 32 mixture components in a gender-independent manner, using all the speech utterances in the TIMIT dataset. Training the UBM, total variability matrix, and the i-vector extraction were done by using the MSR Identity Toolbox via MATLAB [33]. The encoder and decoder of the experimented VAE- and ALI-based networks were configured to have a single hidden layer with 4096 ReLU nodes, and the dimensionality of the latent variables was set to be 200. As depicted in Fig. 4.4, the discriminator network of the ALI-based feature extraction model was configured to follow a similar structure to the multimodal network [44]. The first few hidden layers model the higher-level representation of the GMM supervector and latent variable, and the last hidden layer models the joint information between them. The implementation of the experimented networks was done using Tensorflow [34] and trained using the AdaGrad optimization technique [35]. Also, dropout [36] with a fraction of 0.8 and L2 regularization with a weight of 0.01 were applied for training all the VAEs, and the Baum-Welch statistics extracted from the entire TIMIT dataset were used as training data. A total of 100 samples were used for reparameterization to approximate the expectations in (15).

For all the extracted utterance-level features, linear discriminant analysis (LDA) [15] was applied for feature compensation and the dimensionality was finally reduced to 200. Probabilistic linear discriminant analysis (PLDA) [37] was used for speaker verification, and the speaker subspace dimension was set to be 200.

Two performance measures were evaluated in our experiments: classification error (Class. err.) and EER. The classification error was measured while performing a speaker identification task where each trial utterance was compared with all the enrolled speakers via PLDA, and the enrolled speaker with the highest score was chosen as the identified speaker. Then the ratio of the number of falsely classified samples to the total number of trial samples represents the classification error. The EER is a widely used measure for speaker verification which indicates the error when the false alarm rate (FAR) and the false reject rate (FRR) are the same [15].

4.4.3 Effect of the duration on the latent variable

In order to evaluate the effectiveness of using the latent variable variance as a measure for uncertainty caused by short duration, the differential entropy, which measures the average uncertainty of a random variable, was computed. Since the latent variable $\mathbf{z}(\mathbf{X})$ is assumed to follow a Gaussian distribution, the differential entropy can be formulated as follows:

$$h(\mathbf{z}(\mathbf{X})) = \frac{1}{2}\log(2\pi e)^K + \frac{1}{2}\log\prod_{k=1}^{K}\sigma_k^2(\mathbf{X}). \tag{4.13}$$

In (4.13), K represents the dimensionality of the latent variable and $\sigma_k^2(\mathbf{X})$ is the k^{th} element of $\sigma^2(\mathbf{X})$.

From each speech sample in the entire TIDIGITS dataset, the variance of the 200-dimensional latent variable was obtained using the encoder network and used for computing the differential entropy. As shown in Fig. 4.5, we experimented with the latent variables extracted from three different feature extraction models:

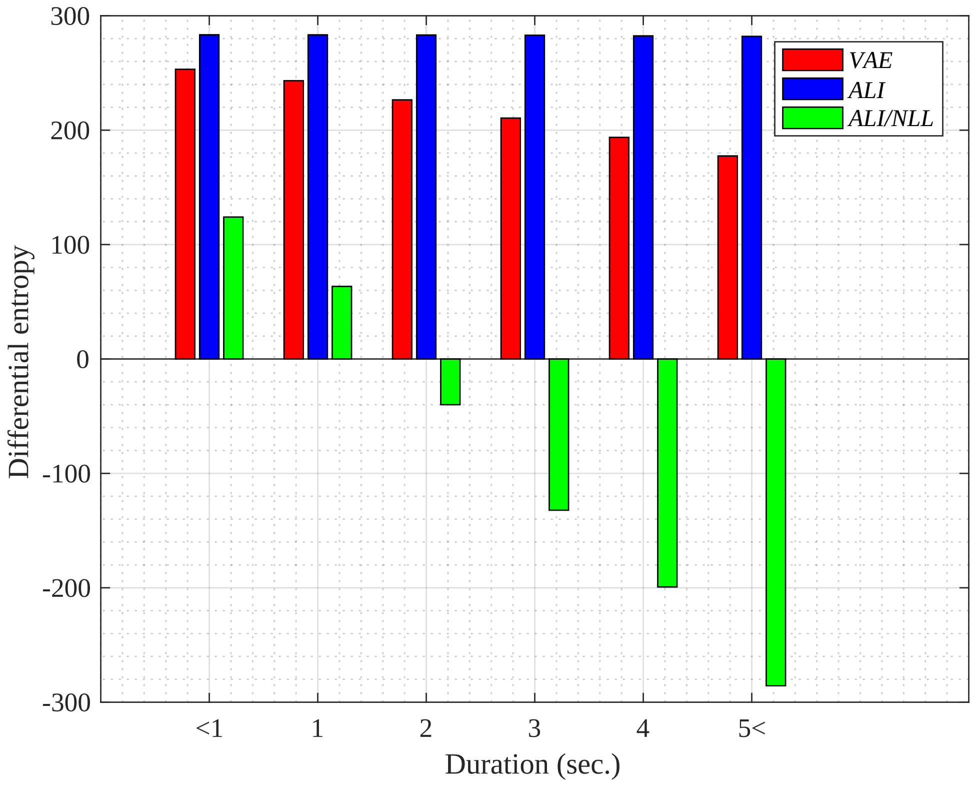


Figure 4.5: Average differential entropy computed using the latent variable variance extracted from the VAE- and ALI-based systems on different durations.

- *VAE*: the VAE-based feature extraction network trained to minimize (4.12),
- *ALI*: the ALI-based feature extraction network trained to minimize the standard GAN objective function (4.3),
- *ALI/NLL*: the proposed ALI-based feature extraction network trained to minimize the negative log-likelihood-based objective function (4.11).

The differential entropies are averaged in 6 different duration groups (i.e. less than 1 second, 1-2 seconds, 2-3 seconds, 3-4 seconds, 4-5 seconds, and more than 5 seconds).

As shown in the result, the differential entropies computed using the variances of the latent variables extracted from *VAE* and *ALI/NLL* gradually decrease as the duration increases. However, the differential entropy computed using the latent variable from *ALI* does not decrease dramatically compared to the other methods. This may be due to the fact that *ALI* is trained only to generate a GMM supervector similar to the one obtained via MAP adaptation (4.9), which is determined by the input Baum-Welch statistics. Therefore the latent variable of *ALI* will only be able to preserve information needed for reconstructing a deterministic distribution. On the other hand, *VAE* and *ALI/NLL* is trained to generate a GMM distribution according to the maximum likelihood criterion, thus their latent variables may capture more information about the variability within the generated GMM distribution.

Another interesting observation from Fig. 4.5 is that the relative decrement in entropy was much greater in *ALI/NLL* than *VAE*. While the change in entropy is rather conservative in the *VAE* case, where the relative decrement between the first duration group (i.e. less than 1 second) and the sixth duration group (i.e. more than 5 seconds) was 29.91%, the entropy in *ALI/NLL* changed dramatically with a relative decrement of 330.17%. This shows that regularizing the latent variable with a joint discriminator network is more effective than using the KL divergence-based regularization for capturing the uncertainty.

4.4.4 Speaker verification and identification with different utterance-level features

In this subsection, we evaluated the performance of the features extracted from various techniques. More specifically, we compared the performance of the conventional i-vector and the latent variable mean (*LM*) and log-variance (*LV*)) extracted from the VAE- and ALI-based feature extractors (i.e. *VAE*, *ALI*, *ALI/NLL*). In ad-

Table 4.1: EER comparison between various feature-level fusions of the conventional i-vector, mean and log-variance of the latent variables extracted from the VAE- and ALI-based feature extractors [%]

	LM	*LV*	*LM+LV*	*i-vector+LM*	*i-vector+LV*	*i-vector+LM+LV*
i-vector(200)	**3.36**					
i-vector(400)				2.68		
i-vector(600)						2.17
VAE	3.61	4.65	2.03	1.78	1.65	0.97
ALI	4.39	4.56	2.32	1.59	1.55	1.03
ALI/NLL	3.64	3.46	1.91	1.55	**1.51**	**0.94**

dition, we conducted feature-level fusion between different features and evaluated their performance. For feature-level fusion, we simply concatenated the different features together to create a supervector. The i-vector features used in this experiment were:

- *i-vector(200)*: standard 200-dimensional i-vector,
- *i-vector(400)*: standard 400-dimensional i-vector,
- *i-vector(600)*: standard 600-dimensional i-vector,

and the latent variable features were:

- *LM*: 200-dimensional latent variable mean,
- *LV*: 200-dimensional latent variable log-variance.

The fusion features used in this experiment were:

- *LM+LV*: concatenation of the 200-dimensional latent variable mean and the log-variance, resulting in a 400-dimensional vector,

Table 4.2: Classification error comparison between various feature-level fusions of the conventional i-vector, mean and log-variance of the latent variables extracted from the VAE- and ALI-based feature extractors [%]

	LM	*LV*	*LM+LV*	*i-vector+LM*	*i-vector+LV*	*i-vector+LM+LV*
i-vector(200)	12.62					
i-vector(400)			7.67			
i-vector(600)						5.07
VAE	**11.89**	17.78	6.94	5.36	4.99	2.75
ALI	16.62	17.31	8.54	5.10	4.79	2.79
ALI/NLL	12.56	12.38	6.76	**3.97**	4.18	**2.49**

- *i-vector+LM*: concatenation of the 200-dimensional i-vector and the 200-dimensional latent variable mean, resulting in a 400-dimensional vector,
- *i-vector+LV*: concatenation of the 200-dimensional i-vector and the 200-dimensional latent variable log-variance, resulting in a 400-dimensional vector,
- *i-vector+LM+LV*: concatenation of the 200-dimensional i-vector and the 200-dimensional latent variable mean and log-variance, resulting in a 600-dimensional vector.

Tables 4.1 and 4.2 respectively give the EER and classification error results obtained by using these features. As depicted in Fig. 4.6, the latent variable mean vector extracted from the VAE- and ALI-based feature extractors (i.e. *VAE*, *ALI*, *ALI/NLL*) shows promising performance. In particular the performance yielded by the latent variable mean of *VAE* and *ALI/NLL* (i.e. *VAE-LM*, *ALI/NLL-LM*) seem to be comparable to the conventional i-vector. On the other hand, as shown in Fig. 4.7, the latent variable log-variance of *VAE* and *ALI* (i.e. *VAE-LV*, *ALI-LV*) shows

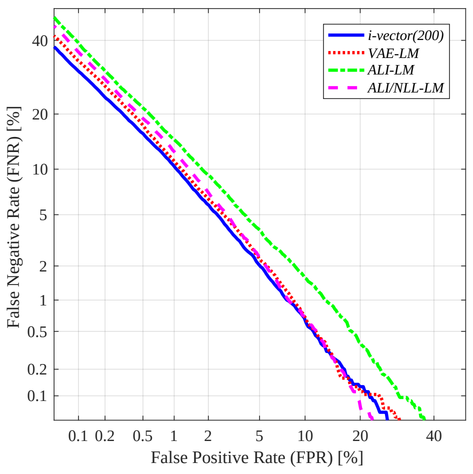


Figure 4.6: DET curves of the speaker verification experiments using latent variable mean as feature.

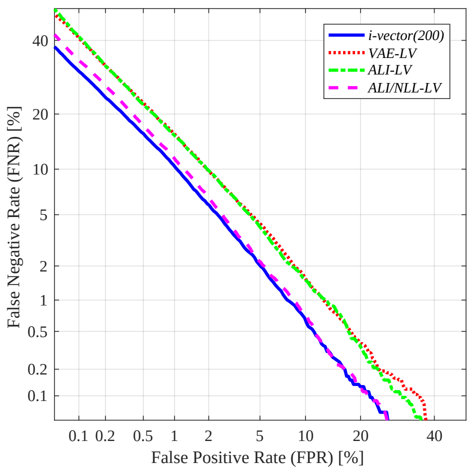


Figure 4.7: DET curves of the speaker verification experiments using latent variable log-variance as feature.

relatively poor performance compared to the latent variable mean features. However, as shown in Fig. 4.7, the latent variable log-variance extracted from *ALI/NLL* (i.e. *ALI/NLL-LV*) outperformed the latent variable log-variance features extracted from all three networks (i.e. *VAE*, *ALI*, *ALI/NLL*) in terms of EER. This shows that the proposed network *ALI/NLL* is capable of generating latent variable variance which not only implies the uncertainty within the input speech but also encodes a sufficient amount of speaker-dependent information. Using the latent variable together (i.e. *LM+LV*) as a combined feature further improved the performance, and the best performing feature was obtained from *ALI/NLL*, which achieved a relative improvement of 28.73% in terms of EER compared to that of *i-vector(400)*. Fig. 4.8 shows the DET curves obtained from the experiments using *LM+LV*.

As shown in Fig. 4.9, augmenting the standard i-vector and the latent variable mean (i.e. *i-vector+LM*) further improved the speaker verification performance. This

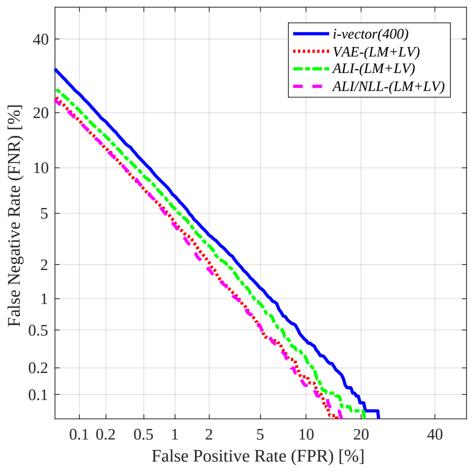


Figure 4.8: DET curves of the speaker verification experiments using the concatenation of the latent variable mean and log-variance as feature.

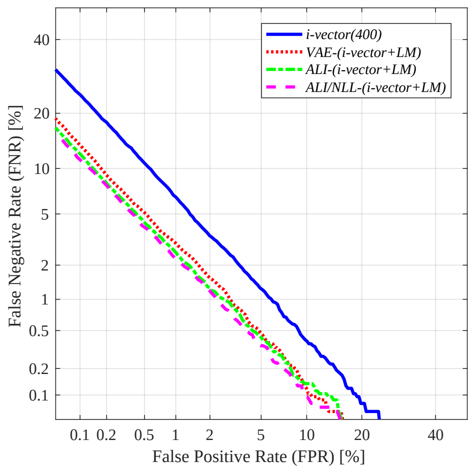


Figure 4.9: DET curves of the speaker verification experiments using the i-vectors augmented with the latent variable mean as feature.

improvement may be attributed to the non-linear feature extraction process of the VAE- and ALI-based methods. Since the latent variable mean is trained to capture the variability within the distribution of the input utterance via a non-linear process, it is likely to encompass information not attainable in the standard i-vector. Especially the one augmented with the latent variable mean extracted from *ALI/NLL* (i.e. *ALI/NLL-(i-vector+LM)*) showed better performance than the ones extracted from *VAE* and *ALI*, achieving a relative improvement of 42.16% in terms of EER compared to *i-vector(400)*. The reason behind this may be due to the fact that the latent variable of *ALI/NLL* can preserve the distinctive information much better by incorporating a joint discriminator instead of regularizing the latent variable distribution via KL divergence.

Likewise, using the i-vector in conjunction with the latent variable log-variance

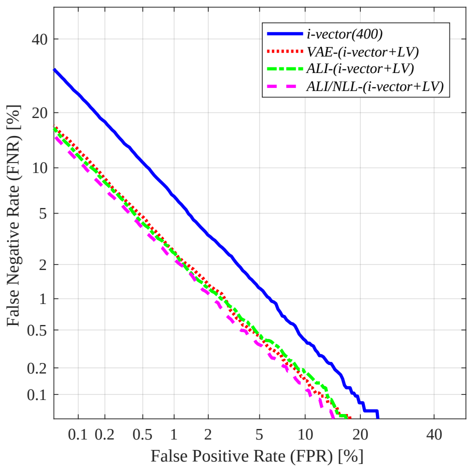


Figure 4.10: DET curves of the speaker verification experiments using the i-vectors augmented with the latent variable log-variance as feature.

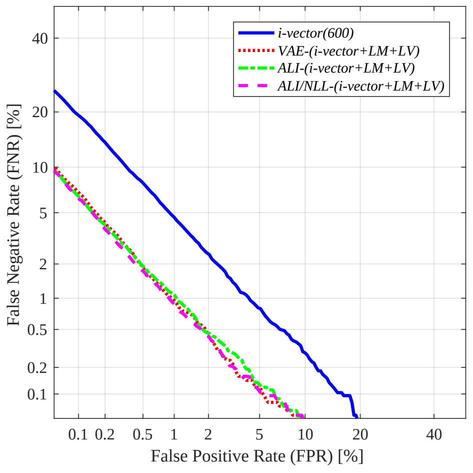


Figure 4.11: DET curves of the speaker verification experiments using the i-vectors augmented with the latent variable mean and log-variance as feature.

(i.e. *i-vector+LV*) also showed improvement in performance. Similar to the *i-vector+LM* experiments, the i-vector augmented with the latent variable log-variance extracted from *ALI/NLL* (i.e. *ALI/NLL-(i-vector+LV)*) outperformed the other methods (i.e. *VAE-(i-vector+LV)*, *ALI-(i-vector+LV)*), achieving a relative improvement of 34.66% compared to *i-vector(400)* in terms of EER. This may be due to the capability of the latent variable variance extracted from *ALI/NLL* in capturing the amount of uncertainty, which has been discussed in the previous subsection. Fig. 4.10 shows the DET curves obtained from the experiments using *i-vector+LV*.

Further improvement was observed when augmented with both the latent variable mean and log-variance (*i-vector+LM+LV*), which can be seen in Fig. 4.11. The standard i-vector used in conjunction with the latent variable mean and log-variance extracted from *ALI/NLL* (i.e. *ALI/NLL-(i-vector+LM+LV)*) showed a relative improvement of 56.68% in terms of EER compared to the conventional i-vector with the same dimension (i.e. *i-vector(600)*).

4.5 Summary

In this Chapter, a novel utterance-level feature extractor using an adversarial learning framework for speaker recognition is proposed. Analogous to the previously proposed VAE-based feature extractor, the architecture proposed in this paper is composed of an encoder and a decoder network where the former estimates the distribution of the latent variable given the speech and the latter generates the GMM from the latent variable. However, in order to prevent the potential loss of distinctive representation for the speaker within the extracted latent variable, the newly proposed feature extractor is trained according to the ALI framework where a joint discriminator network is exploited to ensure that the latent variable and the generated GMM are close to their prior distribution and the GMM obtained through MAP adaptation, respectively.

To evaluate the performance of the features extracted from the proposed system in a short duration scenario, we conducted a set of experiments using the TIDIGITS dataset. From the results, we observed that the variance of the latent variable extracted from the proposed ALI-based feature extractor is more useful to represent the level of uncertainty caused by the short duration of the given speech than the one extracted from the VAE-based feature extractor. Moreover, using the features extracted from the proposed ALI-based method in conjunction with the standard i-vector was shown to be far more effective than the VAE-based method.

Chapter 5

Disentangled speaker and nuisance attribute embedding for robust speaker verification

5.1 Introduction

In recent years, various methods have been proposed utilizing deep learning architectures for extracting embedding vectors and have shown better performance than the i-vector framework when a large amount of training data is available [6]. In [7], a deep neural network (DNN) for frame-level speaker identification was trained and the averaged activation from the last hidden layer, namely, the d-vector, was taken as the embedding vector for text-dependent speaker verification. In [6, 8], a speaker identification model consisting of a frame-level network and a segment-level network was trained and the hidden layer activation of the segment-level network (i.e. x-vector) was extracted as the embedding vector. In [11], long short-term memory (LSTM) layers were adopted to capture the contextual information within the d-vector, and the embedding network was trained to directly optimize the verification score (e.g., cosine similarity) in an end-to-end fashion. The end-to-end d-vector framework was further enhanced in [10] by applying different weight (i.e. attention) to each frame-level activation while obtaining the d-vector, which enables the embedding network to attend more on the frames with relatively higher amount of speaker-dependent information. In [11], a generalized end-to-end loss function, which optimizes the embedding vector to move towards the centroid of the true speaker while departing away from the centroid of the most confusing speaker, was introduced to train the end-to-end d-vector system more efficiently. In [12] and [13], a variational autoencoder (VAE)-based architecture was trained in an unsupervised manner to extract an embedding vector for short-duration speaker verification. Despite their success in well-matched conditions, the deep learning-based embedding methods are vulnerable to the performance degradation caused by mismatched conditions (e.g., channel, noise) [14].

In real life applications, numerous factors can contribute to the mismatches in speaker verification [15]. Especially in forensic situations, channel mismatch often occurs since police officers usually acquire voice recordings using various recording devices (e.g., hidden microphones, mobile phones) [16]. Such variation in recording devices is known to cause variability to the speech distribution, which leads to low speaker identification or verification performance.

Recently, many attempts have been made to extract an embedding vector robust to mismatched conditions. Conventionally, various researches focused on adapting the back-end scoring model (e.g., PLDA) [45] or training the embedding network with an augmented dataset containing various nuisance variability [46]. These methods are proven to be effective when the dataset for the target condition (e.g., noisy evaluation domain) is scarce, but since these methods do not intervene during the embedding extraction, their performance may be bottlenecked by the speaker discriminative capability of the embedding network. Unlike the aforementioned domain adaptation techniques, there have been several methods which aim to directly disentangle the undesired variability while extracting the speaker embeddings. In [14, 23], inspired by the usage of gradient reversal strategy in image classification [26], [47] and robust speech recognition [48, 49], the embedding networks were trained to minimize the speaker classification error while maximizing the error of the subtask (e.g., noise or channel type classification) with the use of gradient reversal layer. Although the gradient reversal strategy has shown meaningful improvement in performance, domain adversarial training using gradient reversal layer is known to be very unstable and sensitive to hyper-parameter setting [50]. In [24], the embedding network was trained to maximize the error of a subtask (i.e. noise type classification) by using an adversarial training strategy similarly to the generative adversarial network (GAN) [43]. The speaker embedding network and the noise classification network are trained competitively; the noise classification network is trained to discriminate the noise type correctly, and at the same time the embedding network is trained to discriminate the speaker while having high uncertainty on the noise type. When training the speaker embedding network, bit-inverted one-hot labels (i.e. anti-labels) were used for noise classification, which would force the embedding network to output a wrong noise label equally. Though the anti-label strategy has proven its strength in noise-robust speaker embedding [24], adversarial training is known to be extremely unstable and difficult [51].

In this paper, we propose a novel approach to disentangle the nuisance attribute information from the speaker embedding vector without the use of gradient reversal or adversarial training. The proposed method employs an embedding network similar to the conventional methods (e.g., d-vector and x-vector). However, unlike the conventional embedding networks, which produce a single embedding vector per utterance, the proposed embedding network simultaneously extracts a speaker- and nuisance attribute-dependent (e.g., recording device-, emotion-dependent) embedding vectors, hence we call the proposed technique joint factor embedding (JFE). In the JFE technique, the embedding network is trained in a fully supervised manner simultaneously with the speaker and nuisance attribute (e.g., channel, emotion) discriminator networks where each discriminator is trained to take the embedding vector as input and identify their respective targets. Analogous to the conventional speaker embedding systems, the proposed embedding network is trained to produce a speaker embedding vector with high speaker discriminability. On the other hand, to disentangle the non-speaker information from the speaker embedding vector, we propose two different ways to increase the nuisance attribute uncertainty inherent in the speaker embedding vector. One way is to train the embedding network to extract a speaker embedding vector to maximize the entropy in nuisance attribute identification, and the other is to decrease the relevancy between the speaker and nuisance embedding vectors by minimizing the mean absolute Pearson's correlation (MAPC) [52].

In order to evaluate the performance of the proposed system in a realistic scenario, we conducted a set of experiments using two datasets:

- RSR2015 Part 3 dataset: a random digits strings speaker verification corpus consisting of speech samples recorded from 6 different hand-held devices [53],

[54].

- VoxCeleb1 dataset: a text-independent speaker verification corpus consisting of speech samples with 8 different emotional states [55].

The experimental results show that the proposed method outperforms the conventional disentanglement methods (i.e. gradient reversal, anti-label) in terms of equal error rate (EER). Moreover, the proposed system performed better than the conventional x-vector on short duration speech samples, which is likely to lack significant phonetic information.

The contributions of this paper are as follows:

- We propose a new method to train a speaker embedding network robust to nuisance attributes, which can be done easily without the use of adversarial training or gradient reversal learning.
- We compared the proposed speaker embedding technique with conventional methods for multi-device and emotional speaker verification.
- We experimented the proposed speaker embedding technique on speech utterances with various durations.

5.2 Joint factor embedding

5.2.1 Joint factor embedding network architecture

Analogous to the conventional disentanglement techniques [14, 23, 24], the proposed method is based on the MTL framework. However, as depicted in Fig. 5.1, unlike the standard MTL embedding system, the embedding network of the proposed framework extracts two different embedding vectors simultaneously: speaker embedding

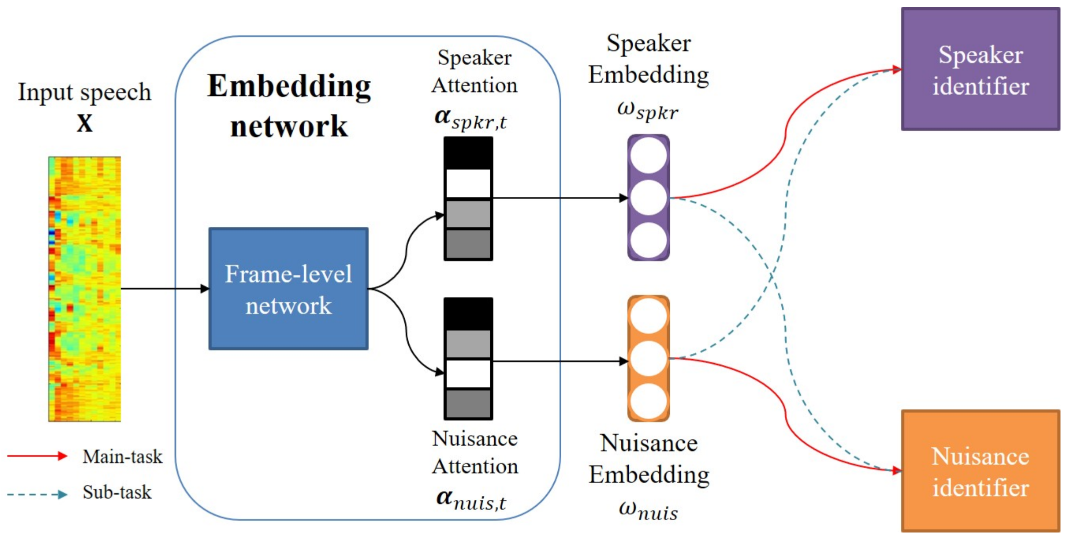


Figure 5.1: The architecture of the proposed joint factor embedding system.

ω_{spkr} and nuisance embedding ω_{nuis}. The speaker embedding vector ω_{spkr} is trained to be dependent solely on the speaker variability while the nuisance embedding vector ω_{nuis} is trained to be dependent on the nuisance (e.g., channel, emotion) variability only. When obtaining ω_{spkr} and ω_{nuis}, different weights are used for aggregating the frame-level outputs as

$$\omega_{spkr} = \sum_{t=1}^{T} \alpha_{spkr,t}\mathbf{h}_t, \tag{5.1}$$

$$\omega_{nuis} = \sum_{t=1}^{T} \alpha_{nuis,t}\mathbf{h}_t \tag{5.2}$$

where $\alpha_{spkr,t}$ and $\alpha_{nuis,t}$ are the speaker and nuisance weights for attention, respectively, which are obtained as in (2.7). The reason why we use separate attention weights for obtaining ω_{spkr} and ω_{nuis} is that we assume that frames with high speaker-dependent information are not always guaranteed to have high nuisance attribute-dependent information. For instance, speaker-dependent information will be high on speech frames, while channel-dependent information will be rather consistent across all frames since even non-speech frames are affected by the recording channel. Once the embedding vectors are extracted, both ω_{spkr} and ω_{nuis} are fed into the speaker and nuisance classification networks.

5.2.2 Training for joint factor embedding

Discriminative training

As described in Table 5.1, the embedding vectors ω_{spkr} and ω_{nuis} are trained with different main task and subtask specifications. In order to maximize the discriminability on their main tasks, the following cross-entropy loss functions are minimized:

$$\mathbf{L}_{s-s,CE} = -\sum_{n=1}^{N} \mathbf{y}_n \log \tilde{\mathbf{y}}_n(\omega_{spkr}), \tag{5.3}$$

$$\mathbf{L}_{c-c,CE} = -\sum_{m=1}^{M} \mathbf{r}_m \log \tilde{\mathbf{r}}_m(\omega_{nuis}). \tag{5.4}$$

By minimizing (5.3) and (5.4) simultaneously, the embedding network is trained to produce ω_{spkr} with high speaker-dependent information and ω_{nuis} with high nuisance attribute-dependent information. Moreover, the attention weights $\alpha_{spkr,t}$ and $\alpha_{nuis,t}$ will be trained to focus on the frames with more meaningful information on their main tasks.

Table 5.1: Main tasks and subtasks for the embedding vectors of the joint factor embedding scheme.

	Main task	Subtask
ω_{spkr}	Speaker classification	Nuisance classification
ω_{nuis}	Nuisance classification	Speaker classification

Disentanglement training

In this paper, we propose two types of loss functions to perform disentanglement in the subtasks of the embedding vectors ω_{spkr} and ω_{nuis}. One way for disentanglement is to directly maximize the entropy (or uncertainty) on their subtasks while training. For ω_{spkr} and ω_{nuis}, the entropies [56] on their subtasks can be computed as

$$\mathbf{L}_{s-c,E} = -\sum_{n=1}^{N} \tilde{\mathbf{y}}_n(\omega_{nuis})\log\tilde{\mathbf{y}}_n(\omega_{nuis}), \tag{5.5}$$

$$\mathbf{L}_{c-s,E} = -\sum_{m=1}^{M} \tilde{\mathbf{r}}_m(\omega_{spkr})\log\tilde{\mathbf{r}}_m(\omega_{spkr}). \tag{5.6}$$

By maximizing (5.5) and (5.6), the uncertainty of the outputs in the subtasks will be maximized, leading the conditional distribution of the subtask classes to approach uniform.

Another way to perform disentanglement is to regularize the embedding vectors ω_{spkr} and ω_{nuis} so as to have low correlation instead of directly maximizing the uncertainty on their subtasks. This can be achieved by maximizing the negative MAPC [52], which can be computed across the mini-batch by

$$\mathbf{L}_{nMAPC} = -\frac{1}{F}\sum_{f=1}^{F} \frac{|cov(\omega_{spkr,f}, \omega_{nuis,f})|}{std(\omega_{spkr,f})std(\omega_{nuis,f})} \tag{5.7}$$

where *cov* is the covariance, *std* is the standard deviation, and F, $\omega_{spkr,f}$, $\omega_{nuis,f}$ are the dimensionality of the embedding vectors, f^{th} element of ω_{spkr} and ω_{nuis}, respectively. Since zero correlation indicates that the two variables are not related, by minimizing the MAPC between ω_{spkr} and ω_{nuis}, the relevancy between the two embedding vectors can be reduced.

The proposed JFE system is trained by simultaneously minimizing the discriminative losses (i.e. cross-entropy) depicted in (5.3) and (5.4), while maximizing the disentanglement loss in (5.5), (5.6), (5.7). In short, the embedding network is trained to minimize the following loss function:

$$\begin{aligned}\mathbf{L}_{JFE} =&\mathbf{L}_{s-s,CE} + \mathbf{L}_{c-c,CE} \\ &- \mathbf{L}_{s-c,E} - \mathbf{L}_{c-s,E} - \mathbf{L}_{nMAPC}.\end{aligned} \tag{5.8}$$

By optimizing the JFE network, the speaker embedding vector ω_{spkr} is trained to be speaker discriminative while having high uncertainty on the nuisance attribute, and the nuisance embedding vector ω_{nuis} aims to be nuisance attribute discriminative while having high uncertainty on the speaker.

5.3 Experiments

5.3.1 Channel disentanglement experiments

Database

In order to evaluate the performance of the proposed technique for a real-life application of speaker verification where multiple recording devices are involved for enrollment and testing, a set of experiments were conducted based on the RSR2015 dataset [53], [54], which is a speaker verification dataset recorded using 6 different hand-held devices (i.e. 1 Samsung Nexus, 2 Samsung Galaxy S, 1 HTC Desire, 1 Samsung Tab, 1 HTC Legend). For training the embedding networks, we used the *background* and *development* subsets of the RSR2015 dataset Part 3, consisting of utterances (recorded from all six devices) spoken by 194 speakers (100 male and 94 female speakers).

The evaluation was performed according to the RSR2015 Part 3 (random digits string) protocol [54] where 106 speakers (57 male and 49 female speakers) are involved. From the RSR2015 Part 3 evaluation dataset, the 10-digits strings of sessions 1, 4, 7 were used for enrollment and the 5-digits strings of sessions 2, 3, 5, 6, 8, 9 were used for testing.

Experimental Setup

To investigate the effects of the proposed JFE strategy on different embedding architecture, two types of frameworks were used for embedding extraction: d-vector and x-vector. For the d-vector-based systems, a single 512-dimensional unidirectional LSTM layer with a projection layer [57] (projected to 256-dimension) was used. By aggregating the LSTM outputs via a weighted average as described in (2.6), 256-dimensional embedding vectors were obtained. Each classification networks (i.e. speaker and channel identifier) consisted of a single 256-dimensional rectified linear unit (ReLU) hidden layer and a softmax output layer where the output size corresponds to the number of speakers or devices within the training set (e.g., 194-dimensional softmax output for speaker classifier and 6-dimensional softmax output for channel classifier). The acoustic features used in the d-vector-based systems were 19-dimensional Mel-frequency cepstral coefficients (MFCCs) and the log-energy extracted at every 10 ms, using a 20 ms Hamming window. Together with the delta and delta-delta of the 19-dimensional MFCCs and the log-energy, the frame-level feature used in our experiments was a 60-dimensional vector.

For the x-vector-based systems, 5 TDNN layers were used as the frame-level network as in the Kaldi x-vector recipe [6]. The frame-level output of the last TDNN layer were aggregated via attention pooling (2.6) and followed by a ReLU layer, resulting in a 512-dimensional embedding vector. The classification networks in the x-vector-based systems consisted of a single 512-dimensional rectified linear unit

(ReLU) hidden layer and a softmax output layer. The acoustic features used in the x-vector-based systems were 30-dimensional MFCCs extracted at every 10 ms, using a 20 ms Hamming window.

The implementation of the embedding systems was done via Tensorflow [34] and trained using the ADAM optimization technique [58] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. All the experimented networks were trained with learning rate 0.001 and batch size 32 for 12,000 iterations. Cosine similarity was used for computing the verification scores in the experiments.

In our experiments, EER was evaluated as the performance measure. The EER indicates the error when the false alarm rate (FAR) and the false reject rate (FRR) are the same.

Comparison between different disentanglement loss terms

In this experiment, we compare the performance of the speaker embeddings obtained from the d-vector-based JFE system trained with different disentanglement loss terms discussed in Section 5.2. The experimented methods are as follows:

- *Only discriminative*: speaker embedding vector extracted from the JFE network trained only with the discriminative loss functions in (5.3) and (5.4) (which is essentially a multi-task learning for the embedding network to encode speaker and nuisance discriminative information),

- *Entropy*: speaker embedding vector extracted from the JFE network trained with the discriminative loss functions in (5.3), (5.4) and the entropy-based disentanglement losses in (5.5) and (5.6),

- *nMAPC*: speaker embedding vector extracted from the JFE network trained with the discriminative loss functions in (5.3), (5.4) and the negative MAPC-based disentanglement losses in (5.7),

- *Entropy + nMAPC*: speaker embedding vector extracted from the JFE network trained with the discriminative loss functions in (5.3), (5.4) and both the entropy-based and the negative MAPC-based disentanglement losses in (5.5), (5.6) and (5.7).

Table 5.2 gives the EER results obtained by using these embeddings. As shown in the results, the embedding extracted from the JFE networks trained with either *Entropy* or *nMAPC* for disentanglement greatly improved the performance compared to *Only discriminative*, which is essentially a standard MTL embedding technique. This implies that both *nMAPC* and *Entropy* are capable of training the embedding network to produce speaker embedding vectors disentangled from non-speaker factors. Especially the *nMAPC* showed relative improvement of 17.99% compared to *Only discriminative*. The best verification performance was achieved by using both disentanglement loss terms (i.e. *Entropy + nMAPC*), yielding a relative improvement of 25.27% in terms of EER. From this, we could assume that *nMAPC* and *Entropy* are useful for disentangling the channel variability from the speaker embedding.

Training Analysis

In order to check if the training scheme of the proposed JFE system achieves our objective (i.e. maximizing the speaker discriminability and channel uncertainty in ω_{spkr}), we analyzed the training loss described in (5.3)-(5.6) of the d-vector-based

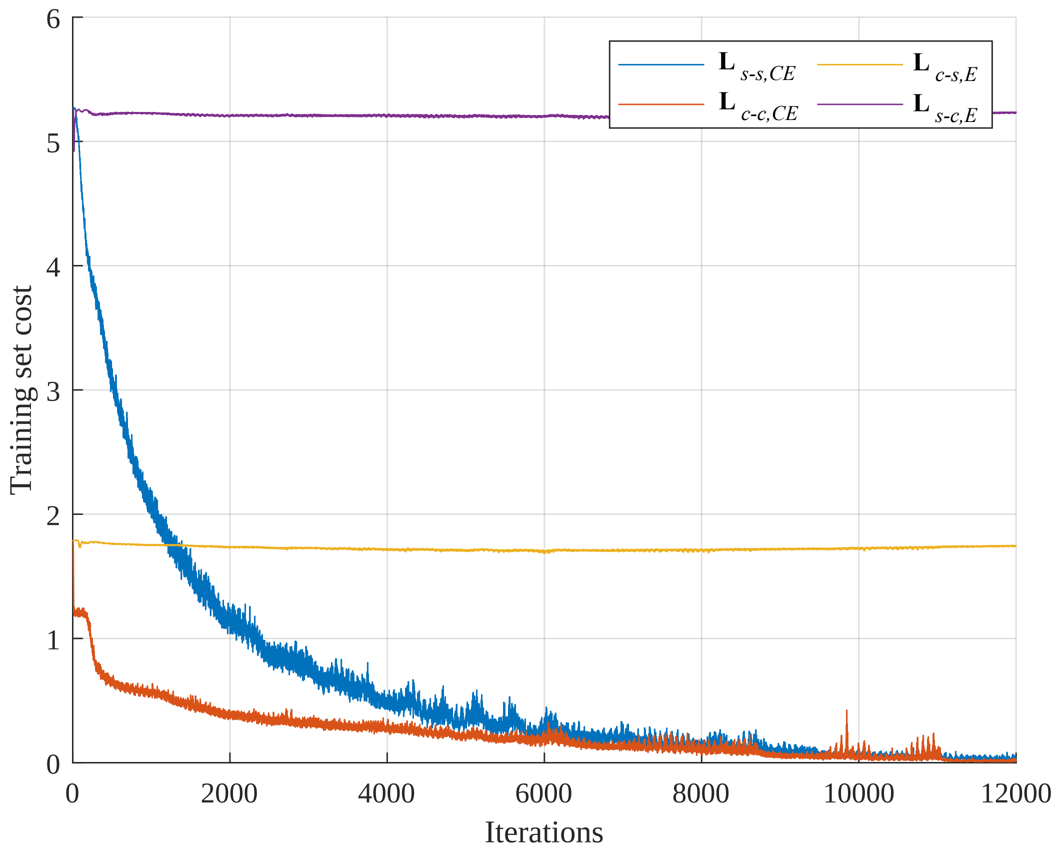
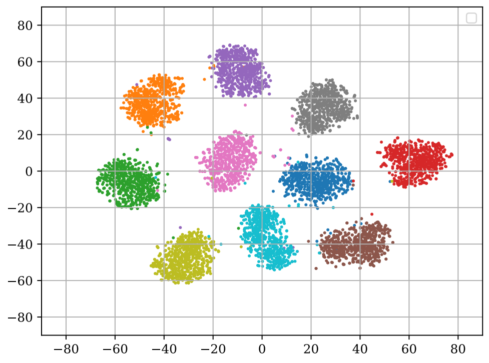
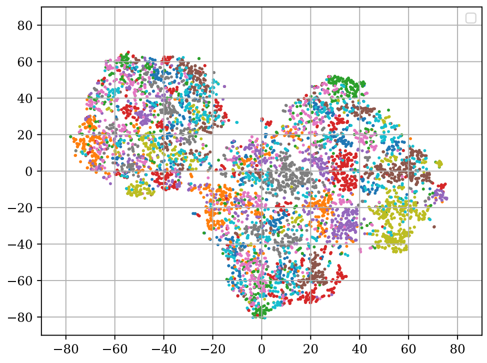


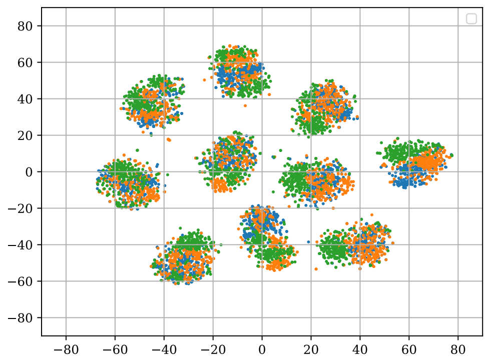
Figure 5.2: The joint factor embedding training loss values on each iteration.



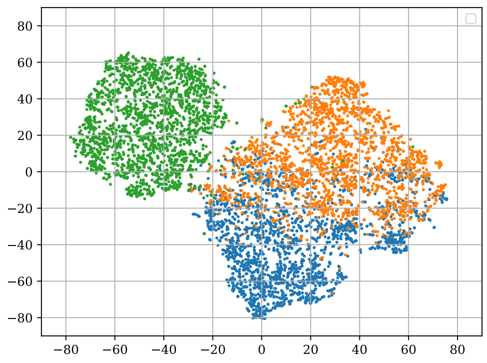
(a)



(b)

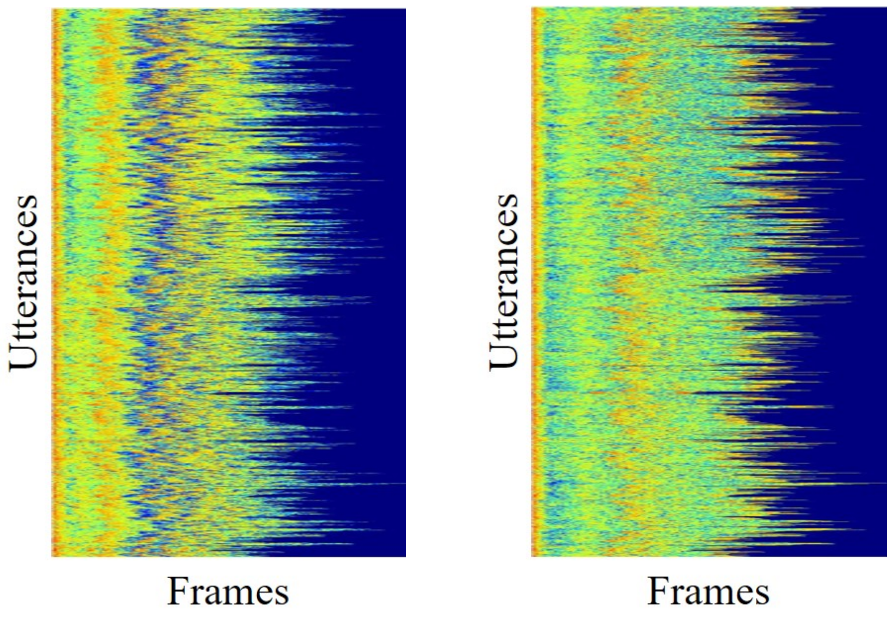


(c)



(d)

Figure 5.3: t-SNE plot of the speaker and channel embedding vectors extracted from 10 speakers and 3 devices. (a) and (c) are the t-SNE plots of the speaker embedding vectors, and (b) and (d) are the t-SNE plots of the channel embedding vectors. Different colors in (a) and (b) indicate different speakers, and different colors in (c) and (d) indicates different devices.



(a) (b)

Figure 5.4: Attention weights of *d-vector (JFE)* for utterances speaking the sentence "only lawyers love millionaires". (a) Attention weights for the speaker embedding vector. (b) Attention weights for the channel embedding vector.

JFE system. As shown in Fig. 5.2, due to the large difference in the unique number of speakers and devices (i.e. 194 speakers and 6 devices), the initial values for $\mathbf{L}_{s-s,CE}$ and $\mathbf{L}_{s-c,E}$ were higher than $\mathbf{L}_{c-c,CE}$ and $\mathbf{L}_{c-s,E}$. The cross-entropy losses (i.e. $\mathbf{L}_{s-s,CE}$ and $\mathbf{L}_{c-c,CE}$) decreased quickly toward 0 when the training iteration increases. On the other hand, the entropy losses (i.e. $\mathbf{L}_{s-c,E}$ and $\mathbf{L}_{c-s,E}$) stayed near at their initial values throughout the training. This indicates that the proposed training scheme increases the discriminability of the speaker and channel embeddings on their main tasks while keeping their uncertainty on the subtasks high as expected.

In Fig. 5.3, the t-SNE plots [59] of the speaker and channel embedding vectors of 10 speakers and 3 devices are shown. As can be seen in Figs. 5.3a and 5.3c, the speaker embedding vectors ω_{spkr} were well separated between different speakers but were highly overlapped when it comes to different devices. Meanwhile, as shown in Figs. 5.3b and 5.3d, the channel embedding vectors ω_{chan} were separately distributed in terms of the device, while they were inseparable in terms of speakers. This confirms that the embedding vectors extracted from the proposed JFE system are discriminative on their main tasks, but are invariant with respect to their subtasks.

Moreover, in Fig. 5.4, the attention weights for the utterance speaking the sentence "only lawyers love millionaires" (i.e. 1^{st} sentence of the RSR2015 Part1 dataset) are shown. It is interesting to see that the difference between speaker attention weights α_{spkr} across the frames were quite dramatic, which indicates that α_{spkr} are likely to attend to certain frames. On the other hand, the channel attention weights α_{chan} were relatively consistent across all frames. These results strongly support our assumption that the frames with high speaker-dependent information are concentrated on specific frames while channel-dependent information is similar across the speech segment.

Comparison between the joint factor embedding scheme and conventional disentanglement methods

In this experiment, we compared the embedding vectors obtained from the proposed joint factor embedding scheme, with those obtained from the conventional disentanglement techniques discussed in Section 2.2. The experimented training strategies are as follows:

- *Softmax*: embedding extracted from an embedding network trained with softmax objective in (2.9),
- *Gradient reversal*: embedding extracted from an embedding network trained with gradient reversal strategy as described in (2.12) where λ was set to be 0 in the beginning and linearly increased every iteration, reaching 1 at the end of the training as in [48],
- *Anti-loss*: embedding extracted from an embedding network trained with anti-loss as described in (2.16) using the same adversarial training strategy described in [24],
- *JFE (proposed)*: speaker embedding extracted from the proposed JFE system trained with the discriminative loss functions in (5.3) and (5.4) and both the entropy-based as shown in (5.5) and (5.6) and the negative MAPC-based disentanglement losses in (5.7).

Table 5.3 show the performance of the d-vector and x-vector-based systems trained with the methods described above. From the results, it could be seen that the overall performance in female speaker was worse than the male speaker. This may be attributed to the fact that the MFCC features are known to be less beneficial in female speech since it does not sufficiently capture the spectral characteristics in high formant frequency region [61]. Generally, the *Anti-loss* disentanglement strategy has shown performance enhancement, achieving a relative improvement of 35.39% in terms of EER in the d-vector-based experiment. On the other hand, *Gradient reversal* method, showed only slightly improved or worse performance over *softmax*. Meanwhile, the speaker embedding extracted from the proposed JFE scheme yielded the best performance in all architectures (i.e., d-vector and x-vector), achieving a relative improvement of 18.39% in EER compared to that of *d-vector (softmax)*. This indicates that the proposed JFE system is capable of disentangling complicated corruptions (i.e. corruption via channel) introduced by different recording devices.

In addition, Table 5.4 show the performance comparison between the state-of-the-art embedding techniques for random digit strings speaker verification (i.e., *DNN i-vectors* and *Uncertainty normalized HMM/i-vector*) [60] and the x-vector-based embedding network trained with the proposed JFE scheme. As shown in the results, *Uncertainty normalized HMM/i-vector* performs better than the *x-vector (softmax)* by a large margin. This is mainly attributed to the fact that the *Uncertainty normalized HMM/i-vector* is trained to model the within-digit variability and scored with prior knowledge on the set of digits being uttered within the test set. Therefore it is not surprising that the *x-vector (softmax)* performs worse than the HMM/i-vector system, since it is trained and evaluated with no information on the context. However, despite the innate disadvantage of the x-vector framework in random digits strings speaker verification, the proposed *x-vector (JFE)* outperformed the *Uncer-*

tainty normalized HMM/i-vector with an relative improvement of 46.05% in terms of male trial EER.

Device disentanglement in domain-mismatch scenario

In this experiment, we compared the performance of the conventional x-vector and the proposed JFE system in a cross-domain text-independent speaker verification scenario. More specifically, both embedding systems were trained using the entire RSR2015 dataset and evaluated on the VoxCeleb1 evaluation subset, which is a dataset collected from Youtube videos recorded from a wide variety of channel and environmental conditions (e.g., videos shot on hand-held devices, interviews from red carpets).

As depicted in Table 5.5, the embeddings extracted from systems trained with RSR2015 showed severe performance degradation. Such degradation was likely caused by the vast variety of channel and environmental conditions within the VoxCeleb1, which are known to cause high within-speaker variability of the extracted speaker embedding vectors. Although the RSR2015 dataset is recorded from multiple different devices, the number of recording devices is limited (i.e. 6 devices) and the speech samples are relatively noise-free since they were recorded in an office environment [53, 54]. Therefore training the embedding system using only the RSR2015 dataset may be insufficient to tackle the challenging condition of the VoxCeleb1 evaluation set. Hence the *x-vector* system trained only for speaker discrimination using RSR2015 showed a relative decrement of 94.83% in terms of EER compared to the network trained with the VoxCeleb1 training set. On the other hand, the degredation of the *JFE* system trained to disentangle the device factor from the speaker embedding was 71.55%, which outperformed the *x-vector* trained with the same dataset with a relative improvement of 11.95%. This indicates that even in a domain-mismatch scenario, the proposed *JFE* is able to alleviate the performance degradation caused by recording device variability.

5.3.2 Emotion disentanglement

Emotion variability can cause severe performance degradation in speaker recognition [62], but emotion disentanglement has not been investigated as much as other nuisance attributes, such as noise or channel distortion. This may be due to the challenging nature of emotion disentanglement since unlike noise or channel, emotional variability is caused by the speaker's vocal tract, which also creates speaker variability. In this subsection, we apply the proposed JFE framework for disentangling the variability induced by the speaker's emotional state.

Dataset

In order to evaluate the performance of the proposed technique for emotion disentanglement, a set of experiments were conducted based on the VoxCeleb1 dataset [55] and the emotion labels provided by the EmoVoxCeleb teacher system [63] [1]. For training the embedding networks, we used the *development* subset of the VoxCeleb1 dataset, consisting of 148,642 utterances collected from 1,211 speakers. According to the emotion labels in EmoVoxCeleb, total 8 emotions are observed in the VoxCeleb1 dataset (i.e., neutral, happy, surprise, sad, angry, disgust, fear, contempt).

The evaluation was performed according to the original VoxCeleb1 trial list, which consists of 4,874 utterances spoken by 40 speakers. The duration of the trial

[1]The emotion labels provided by the EmoVoxCeleb teacher system can be downloaded from here: http://www.robots.ox.ac.uk/ vgg/research/cross-modal-emotions/.

utterances was between 3.97 seconds and 69.05 seconds.

Experimental Setup

The acoustic features used in the experiments were 30-dimensional MFCCs extracted at every 10 ms, using a 20 ms Hamming window. The embedding networks are trained with segments consisting of 250 frames, using the ADAM optimization technique.

For the baseline x-vector framework and joint factor embedding system, 5 TDNN layers were used as the frame-level network according to the Kaldi x-vector recipe [6]. The TDNN outputs are aggregated as described in (2.6), and fed into the utterance-level classification network (i.e. speaker and emotion identifier). Each utterance-level classification network consisted of two 512-dimensional LeakyReLU hidden layers and a softmax output layer where the output size corresponds to the number of speakers or emotions within the training set. All the experimented networks were trained with learning rate 0.001 and batch size 256 for 74,321 iterations. Cosine similarity was used for computing the verification scores in the experiments.

Comparison between the joint factor embedding scheme and conventional embedding techniques

In this experiment, we compare the embedding vectors obtained from the proposed joint factor embedding scheme and the conventional x-vector framework along with techniques reported in recent studies including VGG-M, ResNet-34 and end-to-end verification systems [64, 65]. The experimented methods are as follows:

- *i-vector* [64]: the i-vector performance reported in [64],

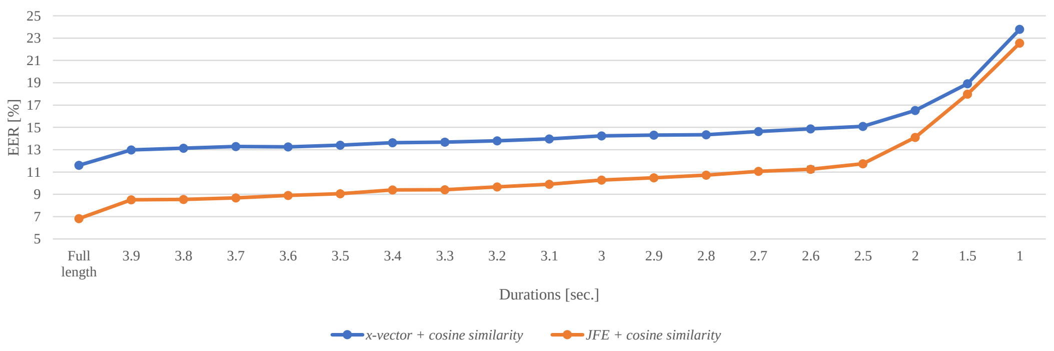


Figure 5.5: EER performance of the proposed joint factor embedding scheme and conventional x-vector on different duration utterances.

- *VGG* [64]: the performance of the embedding extracted from VGG-M, which is a CNN architecture known to perform well on image and speaker classification, reported in [64],

- *Generalized end-to-end* [65]: the performance of the ResNet-34-based end-to-end speaker verification system trained with the generalized end-to-end loss (2.11) reported in [64],

- *All-speaker hard negative mining end-to-end* [65]: the performance of the ResNet-34-based end-to-end speaker verification system trained with the all-speaker hard negative mining loss, which is a modified version of the softmax loss for robust verification, reported in [64],

- *x-vector (softmax)* [64]: the x-vector performance reported in [64],

- *x-vector (our implementation)*: the performance of our implementation of *x-vector (softmax)*,

- *CNN-embedding* [64]: the performance of the embedding extracted from a

CNN-based architecture reported in [64],

- *x-vector (JFE)*: the performance of the speaker embedding extracted from the proposed JFE system trained to disentangle the emotional factor using loss functions (5.3)–(5.7).

As shown in Table 5.6, the proposed *JFE* outperformed the conventional methods with both cosine similarity and PLDA backends. Especially when using PLDA as backend, the *JFE* achieved a relative improvement of 8.16% compared to the *x-vector (our implementation)* in terms of EER. Moreover, training the *JFE* with augmented training data described in [64] (i.e., noise and reverberation augmentation) further improved the performance. The results demonstrate that although the proposed *JFE* is composed of a simple x-vector-like network, it can provide embedding with higher speaker discriminative information than the systems with more complicated architecture.

In addition, we evaluated the conventional x-vector framework and the proposed joint factor embedding scheme on short duration speech samples. Each evaluation was done using randomly truncated trial utterances and the average EERs computed over three evaluations for each duration group are depicted in Fig. 5.5. As shown in the results, both the performance of the joint factor embedding framework and the conventional x-vector were degraded as the duration decreased. This may be due to the lack of phonetically informative frames since a critical amount of speaker relevant information is contained in the phonetic characteristics [19]. However, the emotion disentangled speaker embedding obtained by the proposed *JFE* outperformed the conventional *x-vector* even with short duration speech segments.

5.3.3 Noise disentanglement

In this subsection, we apply the proposed JFE framework for disentangling the noise variability.

Dataset

In order to evaluate the performance of the proposed technique for noise disentanglement, we conducted an experiment based on the SiTEC dataset [66], where each utterance was mixed with 4 different background noises (i.e., restaurant, office, cafeteria, construction) from the ITU-T P.501 dataset [67] on 3 different SNR conditions (i.e., 0, 5, 10dB). For training the embedding networks, we used 187,470 utterances from 300 speakers which were mixed with cafeteria and restaurant noises .

The evaluation was performed based on the 2,000 utterances from 100 speakers. The evaluation set was split into two, where the first half was used as enrollment and the rest was used for trial. The enrollment set was mixed only with office noise, while the trial set was mixed with construction noise.

Experimental Setup

The acoustic features used in the experiments were 30-dimensional MFCCs extracted at every 10 ms, using a 20 ms Hamming window. The embedding networks are trained with segments consisting of 250 frames, using the ADAM optimization technique.

For the baseline x-vector framework and joint factor embedding system, 5 TDNN layers were used as the frame-level network according to the Kaldi x-vector recipe [6]. The TDNN outputs are aggregated as described in (2.6), and fed into the utterance-level classification network (i.e. speaker and emotion identifier). Each utterance-level classification network consisted of two 512-dimensional LeakyReLU hidden layers and a softmax output layer where the output size corresponds to the number of speakers or emotions within the training set. All the experimented networks were trained with learning rate 0.001 and batch size 256 for 74,321 iterations. Cosine similarity was used for computing the verification scores in the experiments.

Comparison between the joint factor embedding and x-vector

In this experiment, we compare the embedding vectors obtained from the proposed joint factor embedding scheme and the conventional x-vector framework. As shown in Table 5.7, the proposed *JFE* outperformed the conventional methods in all noise conditions. Especially in the SNR 0dB condition, the *JFE* achieved a relative improvement of 12.93% compared to the *x-vector* in terms of EER. This indicates that the proposed JFE system is capable of disentangling corruptions introduced by different background noises.

5.4 Summary

In this Chapter, a novel approach for extracting an embedding vector robust to variability caused by nuisance attributes for speaker verification is proposed. In order to disentangle the nuisance variability from the speaker embedding vector, we introduce a JFE scheme where two types of embedding vectors are extracted, each dependent solely on the speaker or nuisance attribute, respectively. The proposed JFE network is trained simultaneously with the speaker and nuisance attribute classification networks where the speaker and nuisance embedding vectors are optimized to have good discriminability for their main task while having high uncertainty on their subtask.

To evaluate the performance of the embedding vector extracted from the proposed system in a realistic scenario, we conducted a set of speaker verification experiments using the RSR2015 dataset, which is composed of utterances recorded using multiple different hand-held devices, and VoxCeleb1 dataset, which is composed of various emotional speech utterances. From the results, it is shown that the proposed JFE scheme is capable of obtaining speaker embedding vectors with high speaker discriminability while showing robustness to channel and emotional variability. Moreover, we observed that the proposed embedding vector performs better than the conventional embedding technique with short duration speech segments.

Table 5.2: EER (%) comparison between the speaker embedding vectors extracted from the joint factor embedding networks trained with various disentanglement losses.

Loss	EER [%]
Only discriminative	11.28
Entropy	9.61
nMAPC	9.25
Entropy + nMAPC	**8.43**

Table 5.3: EER (%) comparison between the speaker embedding vectors extracted from the proposed joint factor embedding and the other embedding techniques.

	Objective	EER [%]
d-vector	*Softmax*	10.72
	Gradient reversal	10.37
	Anti-loss	10.47
	JFE (proposed)	**8.43**
x-vector	*Softmax*	2.26
	Gradient reversal	5.87
	Anti-loss	1.46
	JFE (proposed)	**1.07**

Table 5.4: Gender-dependent EER (%) comparison between the speaker embedding vectors extracted from the x-vector-based embedding systems and the state-of-the-art i-vector-based systems.

Methods	EER [%]	
	Male	Female
x-vector (Softmax)	2.09	2.48
DNN i-vectors [60]	1.70	2.69
Uncertainty normalized HMM/i-vector [60]	1.52	1.77
x-vector (GRL)	3.75	4.17
x-vector (Anti-loss)	1.25	1.66
x-vector (JFE)	**0.82**	**1.29**

Table 5.5: EER (%) comparison between the speaker embedding vectors extracted from the proposed joint factor embedding and the conventional x-vector framework evaluated on the VoxCeleb1 evaluation set.

Objective	Training data	EER [%]
x-vector (softmax)	VoxCeleb1	11.6
	RSR2015	22.6
x-vector (JFE)	RSR2015	19.9

Table 5.6: EER (%) comparison between the speaker embedding vectors extracted from the proposed joint factor embedding and the conventional methods.

Methods	Scoring	Data augmentation	EER [%]
i-vector [64]	PLDA	X	8.8
VGG [64]	Cosine similarity	X	7.8
Generalized end-to-end [65]	Cosine similarity	X	10.7
All-speaker hard negative mining end-to-end [65]	Cosine similarity	X	5.6
x-vector (softmax) [64]	Cosine similarity	X	11.3
	PLDA	X	7.1
		O	6.0
x-vector (our implementation)	PLDA	O	4.9
CNN-embedding [64]	Cosine similarity	X	7.3
	PLDA	X	5.9
		O	5.3
x-vector (JFE)	Cosine similarity	X	**6.8**
	PLDA	X	**5.4**
		O	**4.4**

Table 5.7: Comparison between the speaker embedding vectors extracted from the proposed joint factor embedding and the x-vector system.

	Methods	Accuracy [%]	EER [%]	DCF08
SNR 0dB	x-vector	92.90	3.79	0.40
	JFE (proposed)	94.10	3.30	0.35

	Methods	Accuracy [%]	EER [%]	DCF08
SNR 5dB	x-vector	94.90	3.25	0.34
	JFE (proposed)	95.20	2.89	0.27

	Methods	Accuracy [%]	EER [%]	DCF08
SNR 10dB	x-vector	95.30	3.21	0.32
	JFE (proposed)	95.60	2.84	0.22

Chapter 6

Conclusions

This dissertation addresses the limitations of the conventional deep embedding techniques for speaker verification. In order to tackle the problem of utilizing unlabeled datasets for training the deep embedding systems and performance degradation when dealing with speech samples with different conditions, we proposed several embedding methods.

Firstly, we have proposed a variational autoencoder (VAE)-based embedding framework, which extracts the total variability embedding and a representation for the uncertainty within the input speech. Unlike the conventional deep learning-based embedding techniques, the proposed system is trained in an unsupervised manner. From a number of experiments, it has been shown that the proposed method outperforms the conventional i-vector framework in a short duration speaker verification scenario.

Secondly, in order to prevent the potential information loss caused by the Kullback-Leibler divergence regularization term in the VAE-based embedding system, we have proposed an adversarially learned inference (ALI)-based embedding framework. Experimental results show that the proposed method can represent the uncertainty caused by the short duration better than the VAE-based method.

Finally, we proposed a new fully supervised method for extracting a speaker embedding vector disentangled from the variablility caused by the non-speaker nuisance attributes. The proposed framework was compared with the conventional deep learning-based embedding methods, and experimental show that the proposed approach can extract the speaker embeddings robust to channel and emotional variability.

Bibliography

[1] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using gmm supervectors for speaker verification," *IEEE signal processing letters*, vol. 13, no. 5, pp. 308–311, 2006.

[2] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE transactions on speech and audio processing*, vol. 13, no. 3, pp. 345–354, 2005.

[3] N. Dehak, P. Kenny, R. Dehak, O. Glembek, P. Dumouchel, L. Burget, V. Hubeika, and F. Castaldo, "Support vector machines and joint factor analysis for speaker verification," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4237–4240, IEEE, 2009.

[4] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[5] P. Kenny, "A small footprint i-vector extractor," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[6] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5329–5333, IEEE, 2018.

[7] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4052–4056, IEEE, 2014.

[8] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification.," 2017.

[9] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5115–5119, IEEE, 2016.

[10] F. R. rahman Chowdhury, Q. Wang, I. L. Moreno, and L. Wan, "Attention-based models for text-dependent speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5359–5363, IEEE, 2018.

[11] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879–4883, IEEE, 2018.

[12] W. H. Kang and N. S. Kim, "Unsupervised learning of total variability embedding for speaker verification with random digit strings," *Applied Sciences*, vol. 9, no. 8, p. 1597, 2019.

[13] W. H. Kang and N. S. Kim, "Adversarially learned total variability embedding for speaker recognition with random digit strings," *Sensors*, vol. 19, no. 21, p. 4709, 2019.

[14] Z. Meng, Y. Zhao, J. Li, and Y. Gong, "Adversarial speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6216–6220, IEEE, 2019.

[15] J. H. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal processing magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[16] M. I. Mandasari, M. McLaren, and D. A. van Leeuwen, "Evaluation of i-vector speaker recognition systems for forensic application," 2011.

[17] S. Yao, R. Zhou, and P. Zhang, "Speaker-phonetic i-vector modeling for text-dependent speaker verification with random digit strings," *IEICE TRANSACTIONS on Information and Systems*, vol. 102, no. 2, pp. 346–354, 2019.

[18] R. Saeidi and P. Alku, "Accounting for uncertainty of i-vectors in speaker recognition using uncertainty propagation and modified imputation," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[19] T. Hasan, R. Saeidi, J. H. Hansen, and D. A. Van Leeuwen, "Duration mismatch compensation for i-vector based speaker recognition systems," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7663–7667, IEEE, 2013.

[20] M. I. Mandasari, R. Saeidi, M. McLaren, and D. A. van Leeuwen, "Quality measure functions for calibration of speaker recognition systems in various duration conditions," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 11, pp. 2425–2438, 2013.

[21] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[22] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Twelfth annual conference of the international speech communication association*, 2011.

[23] X. Fang, L. Zou, J. Li, L. Sun, and Z.-H. Ling, "Channel adversarial training for cross-channel text-independent speaker recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6221–6225, IEEE, 2019.

[24] J. Zhou, T. Jiang, L. Li, Q. Hong, Z. Wang, and B. Xia, "Training multi-task adversarial network for extracting noise-robust speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6196–6200, IEEE, 2019.

[25] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[26] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural net-

works," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[27] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[28] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[29] R. Salakhutdinov, "Learning deep generative models," *Annual Review of Statistics and Its Application*, vol. 2, pp. 361–385, 2015.

[30] R. Leonard, "A database for speaker-independent digit recognition," in *ICASSP'84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 9, pp. 328–331, IEEE, 1984.

[31] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "Darpa timit acoustic phonetic continuous speech corpus cdrom," 1993.

[32] R. Blouet, P. Duhamel, J. Mariethoz, S. Meigner, A. Ozerov, and J. Prado, "Spro: speech signal processing toolkit." `http://gforge.inria.fr/projects/spro`, 2004.

[33] S. O. Sadjadi, M. Slaney, and L. Heck, "Msr identity toolbox v1.0: A matlab toolbox for speaker recognition research," Tech. Rep. MSR-TR-2013-133, September 2013.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving,

M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[35] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[37] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Twelfth annual conference of the international speech communication association*, 2011.

[38] "The nist year 2008 speaker recognition evaluation plan." `http://www.itl.nist.gov/iad/mig//tests/sre/2008/`, 2008.

[39] "The nist year 2010 speaker recognition evaluation plan." `http://www.itl.nist.gov/iad/mig//tests/sre/2010/`, 2008.

[40] L. Theis, A. v. d. Oord, and M. Bethge, "A note on the evaluation of generative models," *arXiv preprint arXiv:1511.01844*, 2015.

[41] W. Shang, K. Sohn, Z. Akata, and Y. Tian, "Channel-recurrent variational autoencoders," *arXiv preprint arXiv:1706.03729*, 2017.

[42] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," *arXiv preprint arXiv:1606.00704*, 2016.

[43] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[44] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *ICML*, 2011.

[45] X. Wang, L. Li, and D. Wang, "Vae-based domain adaptation for speaker verification," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 535–539, IEEE, 2019.

[46] S. Latif, R. Rana, S. Khalifa, R. Jurdak, J. Qadir, and B. W. Schuller, "Deep representation learning in speech processing: Challenges, recent advances, and future trends," *arXiv preprint arXiv:2001.00378*, 2020.

[47] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*, pp. 1180–1189, PMLR, 2015.

[48] Y. Shinohara, "Adversarial multi-task learning of deep neural networks for robust speech recognition.," 2016.

[49] A. Tripathi, A. Mohan, S. Anand, and M. Singh, "Adversarial learning of raw speech features for domain invariant speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5959–5963, IEEE, 2018.

[50] Y. Zhang, R. Weiss, H. Zen, Y. Wu, Z. Chen, R. Ryan-Skerry, Y. Jia, A. Rosenberg, and B. Ramabhadran, "Learning to speak fluently in a foreign language: multilingual speech synthesis and cross-language voice cloning," in *Interspeech*, 2019.

[51] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv preprint arXiv:1701.04862*, 2017.

[52] O. Mogren, *Representation learning for natural language.* PhD thesis, 2018.

[53] A. Larcher, K. A. Lee, B. Ma, and H. Li, "Rsr2015: Database for text-dependent speaker verification using multiple pass-phrases," in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[54] A. Larcher, K. A. Lee, B. Ma, and H. Li, "Text-dependent speaker verification: Classifiers, databases and rsr2015," *Speech Communication*, vol. 60, pp. 56–77, 2014.

[55] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[56] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in neural information processing systems*, pp. 2234–2242, 2016.

[57] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[58] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[59] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[60] N. Maghsoodi, H. Sameti, H. Zeinali, and T. Stafylakis, "Speaker recognition with random digit strings using uncertainty normalized hmm-based i-vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1815–1825, 2019.

[61] X. Zhou, D. Garcia-Romero, R. Duraiswami, C. Espy-Wilson, and S. Shamma, "Linear versus mel frequency cepstral coefficients for speaker recognition," in *2011 IEEE Workshop on Automatic Speech Recognition Understanding*, pp. 559–564, 2011.

[62] L. Chen and Y. Yang, "Emotional speaker recognition based on model space migration through translated learning," in *Chinese Conference on Biometric Recognition*, pp. 394–401, Springer, 2013.

[63] S. Albanie, A. Nagrani, A. Vedaldi, and A. Zisserman, "Emotion recognition in speech using cross-modal transfer in the wild," in *Proceedings of the 26th ACM international conference on Multimedia*, pp. 292–301, 2018.

[64] S. Shon, H. Tang, and J. Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 1007–1013, IEEE, 2018.

[65] H.-S. Heo, J.-w. Jung, I.-H. Yang, S.-H. Yoon, H.-j. Shim, and H.-J. Yu, "End-to-end losses based on speaker basis vectors and all-speaker hard negative mining for speaker verification," *arXiv preprint arXiv:1902.02455*, 2019.

[66] "The speech information technology and industry promotion center." `http://www.sitec.or.kr/`, 2020.

[67] ITU-T Recommendation P.501, "Test signals for use in telephonometry," tech. rep., 2017.

국문초록

최근 몇년간 다양한 딥러닝 기반 성문 추출 기법들이 제안되어 왔으며, 화자 인식에서 높은 성능을 보였다. 하지만 고전적인 성문 추출 기법에서와 마찬가지로, 딥러닝 기반 성문 추출 기법들은 서로 다른 환경 (e.g., 녹음 기기, 감정)에서 녹음된 음성들을 분석하는 과정에서 성능 저하를 겪는다. 또한 기존의 가우시안 혼합 모델 (Gaussian mixture model, GMM) 기반의 기법들 (e.g., GMM 슈퍼벡터, i-벡터)와 달리 딥러닝 기반 성문 추출 기법들은 교사 학습을 통하여 최적화되기에 라벨이 없는 데이터를 활용할 수 없다는 한계가 있다.

본 논문에서는 variational autoencoder (VAE) 기반의 성문 추출 기법을 제안하며, 해당 기법에서는 음성 분포 패턴을 요약하는 벡터와 음성 내의 불확실성을 표현하는 벡터를 추출한다. 기존의 딥러닝 기반 성문 추출 기법 (e.g., d-벡터, x-벡터)와는 달리, 제안하는 기법은 비교사 학습을 통하여 최적화 되기에 라벨이 없는 데이터를 활용할 수 있다. 더 나아가 VAE의 KL-divergence 제약 함수로 인한 정보 손실을 방지하기 위하여 adversarially learned inference (ALI) 기반의 성문 추출 기법을 추가적으로 제안한다. 제안한 VAE 및 ALI 기반의 성문 추출 기법은 짧은 음성에서의 화자 인증 실험에서 높은 성능을 보였으며, 기존의 i-벡터 기반의 기법보다 좋은 결과를 보였다.

또한 본 논문에서는 성문 벡터로부터 비 화자 요소 (e.g., 녹음 기기, 감정)에 대한 정보를 제거하는 학습법을 제안한다. 제안하는 기법은 화자 벡터와 비화자 벡터를 동시에 추출하며, 각 벡터는 자신의 주 목적에 대한 정보를 최대한 많이 유지하되, 부 목

적에 대한 정보를 최소화하도록 학습된다. 기존의 비 화자 요소 정보 제거 기법들 (e.g., adversarial learning, gradient reversal)에 비하여 제안하는 기법은 휴리스틱한 학습 전략을 요하지 않기에, 보다 안정적인 학습이 가능하다. 제안하는 기법은 RSR2015 Part3 데이터셋에서 기존 기법들에 비하여 높은 성능을 보였으며, 성문 벡터 내의 녹음 기기 및 감정 정보를 억제하는데 효과적이었다.

주요어: 화자 인식, 성문 추출, 화자 인증, 비교사 표현 학습, supervised disentanglement.

학번: 2014-21697