# When Speaker Recognition Meets Noisy Labels: Optimizations for Front-Ends and Back-Ends

Lin Li, *Member, IEEE*, Fuchuan Tong, and Qingyang Hong, *Member, IEEE*

***Abstract*—A typical speaker recognition system often involves two modules: a feature extractor front-end and a speaker identification back-end. Despite the superior performance that deep neural networks have achieved for the front-end, their success benefits from the availability of large-scale, correctly labeled datasets. While label noise is unavoidable in speaker recognition datasets, both the front-end and back-end are affected by label noise, which degrades speaker recognition performance. In this paper, we first conduct comprehensive experiments to help improve our understanding of the effects of label noise on both the front-end and back-end. Then, we propose a simple yet effective training paradigm and loss correction method to handle label noise in the front-end. We combine our proposed method with the recently proposed Bayesian estimation of PLDA for noisy labels, and the whole system shows strong robustness to label noise. Furthermore, we show two practical applications of the improved system: one application corrects noisy labels based on an utterance's chunk-level predictions, and the other algorithmically filters out high-confidence noisy samples within a dataset. By applying the second application to the NIST SRE04–10 dataset and verifying filtered utterances by human validation, we identify that approximately 1% of the NIST SRE04–10 dataset is made up of label errors.**

***Index Terms*—Speaker recognition, noisy labels, x-vector, probabilistic linear discriminant analysis.**

## I. Introduction

SPEAKER recognition is a typical biometric authentication technology that verifies the identities of speakers from their voices. A typical speaker recognition system often involves two modules: a feature extractor front-end and a speaker identification back-end. The front-end extracts low-dimensional discriminative speaker representations (embeddings) from variable-length utterances, whereas the back-end determines whether two embeddings are from the same speaker [1]. The Gaussian Mixture Model Universal Background Model (GMM-UBM) [2] and i-vectors [3] are two typical conventional speaker recognition models [4]. Probabilistic Linear Discriminant Analysis (PLDA) [5]–[8] is commonly used as a back-end scoring model. To satisfy the PLDA Gaussian assumptions for training data [9], extracted features generally require preprocessing, such as Linear Discriminant Analysis (LDA) and length normalization [10], before being used to train a PLDA model. Both LDA and PLDA are trained in a supervised manner that requires training data with corresponding speaker labels.

Along with the increasing amount of training data and the development of neural networks, state-of-the-art performances for speaker recognition have been achieved by deep neural networks [4]. Among these networks, the x-vector [11] front-end is perhaps the most popular deep speaker embedding architecture. The x-vector directly replaces the i-vector to extract discriminative speaker representations using time delay neural network (TDNN) layers [12] with a statistical pooling layer. Based on the x-vector architecture, multiple deep speaker embedding network variants [13], [14] have been proposed to boost recognition performance. In addition, margin-based objective functions [15], [16] have been widely used to learn more discriminative speaker representations. Although these methods have achieved remarkable success, supervised training for deep embedding models requires large-scale datasets that are correctly labeled.

Unfortunately, erroneously labeled samples are unavoidable during speaker utterance collections. This phenomenon is denoted as label noise, and incorrectly labeled utterances are denoted as noisy samples. For instance, the NIST SRE18 [17] development set does not provide speaker labels but instead only provides a phone number corresponding to each utterance [18]. The VoxCeleb dataset [19] is collected from YouTube, and the speaker identities are confirmed through facial recognition based on convolutional neural networks (CNNs). Typically, these noisy labels can be categorized into three categories: *closed-set*, *open-set*, and *mixed-set*. The closed-set refers to noisy samples whose true labels are contained within the training classes. The open-set contains noisy samples whose true labels are outside the training set. The mixed-set refers to mixed closed-set and open-set label noise. Fig. 1 provides a pictorial illustration of noisy labels. Label noise would impair both the front-end and back-end model training, thereby degrading the speaker recognition performance. In this paper, we start with comprehensive explorations of the closed-set, as it is a more challenging scenario [20], and then explore the open-set, and mixed-set label noise.

For a training dataset with an unknown number of noisy samples, the front-end goal is to learn discriminative feature spaces where different speaker embeddings are adequately separated; which is so-called *learning with label noise*. Research on

Manuscript received July 1, 2021; revised January 23, 2022, March 24, 2022, and April 20, 2022; accepted April 20, 2022. Date of publication April 26, 2022; date of current version May 5, 2022. This work was supported in part by the National Natural Science Foundation of China under Grants 61876160 and 62001405, and in part by the Fundamental Research Funds for the Central Universities under Grant 20720210087. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Omid Sadjadi. *(Corresponding authors: Qingyang Hong; Lin Li.)*

Lin Li and Fuchuan Tong are with XMU Speech Lab and the School of Electronic Science and Engineering, Xiamen University, Xiamen 361005, China (e-mail: lilin@xmu.edu.cn; tongfuchuan@stu.xmu.edu.cn).

Qingyang Hong is with XMU Speech Lab and the School of Informatics, Xiamen University, Xiamen 361005, China (e-mail: qyhong@xmu.edu.cn).

Digital Object Identifier 10.1109/TASLP.2022.3169977

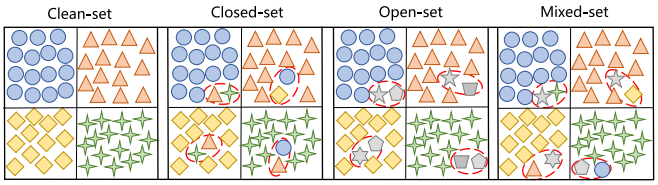



Fig. 1. An illustration of noisy labels. Supposing there are four categories in a clean-set, "closed-set" indicates that some samples have erroneously been given wrong labels from other classes within the dataset; "open-set" means that some out-of-set samples have been grouped into the existing labels within the dataset, and "mixed-set" denotes the conditions of both closed-set and open-set label noise.

learning with label noise is flourishing [21]–[25]. The authors of [26] provide a comprehensive overview of recently proposed approaches for learning with label noise. These methods can be roughly divided into four categories: robust loss functions, robust architecture, robust regularization, and sample selection. A robust loss function [22], [24], [27] prevents a network from overfitting noisy samples by modifying the loss value or designing a more robust objective function. Robust architecture [28], [29] aims to model a noise transition matrix from a noisy dataset by using an auxiliary architecture. Robust regularization [30], [31] reduces the impact of noisy samples by adding regularization techniques. Finally, sample selection [32], [33] handles label noise by selecting correctly-labeled samples.

Meanwhile, learning with label noise has become a popular research topic in the study of speaker recognition. For the x-vector front-end, the detrimental effects of label noise in speaker recognition are confirmed in [34], and the author proposed relaxing the constraints on speaker identity in the entropy loss function to prevent the network from fitting noisy samples. Pham *et al.* [35] conducted extensive experiments to investigate the effects of different types of label noise on the x-vector baseline. The authors of [36] proposed an iterative noisy label detection approach to refine training data labels. For the back-end, Borgström *et al.* [18], [37], [38] proposed a novel method for Bayesian estimation of PLDA when label noise occurs during PLDA training; we refer to this method as NL-PLDA. However, the existing literature either focuses only on the front-end or only on the back-end, and current research does not comprehensively analyze the impact of label noise on different components of speaker recognition systems.

In this paper, we simultaneously optimize the front-end and back-end of speaker recognition systems when training datasets are noisy. For the network front-end, we propose a simple yet effective training paradigm to prevent networks from fitting noisy samples. The proposed framework consists of three major components:

1) A label confidence training scheme that incorporates network-predicted pseudo-labels into the loss function; this method is similar to Bootstrapping [22], but we use a well-designed dynamically increasing confidence weight.

2) A rescaling strategy that reduces the posterior probability of clean labels to emphasize them more in the loss function.

3) An improved AM-Softmax loss function that relaxes the intra-class constraint.

For the back-end, we treat the true labels as multinomial random variables and train an NL-PLDA model to perform speaker identification scoring.

This paper builds upon our previous work on combating noisy labels [39]. Instead of conducting experiments on the VoxCeleb dataset, as in [39], the experiments in this paper are conducted on the Switchboard [40]–[43] and NIST SRE 2004–2010 (SRE04–10 in short) [44] datasets. In addition, more comprehensive experiments are conducted to analyze the effects of label noise on the x-vector, LDA, PLDA, and NL-PLDA models by setting different label error rates in closed-set, open-set, and mixed-set scenarios. The contribution of this paper are multifold and are summarized as follows:

1) In the front-end, a label confidence training paradigm with a dynamic confidence policy, a rescaling strategy, and an improved AM-Softmax are proposed for front-end learning with label noise. In combination with these components, the network shows consistent improvements in the robustness to label noise. Furthermore, a label correction method based on chunk-level label predictions is proposed that significantly reduces the number of noisy samples in a dataset.

2) In the back-end, we show how to apply NL-PLDA to filter out noisy labels. For further practical contribution, we provide an optimized expectation-maximization (EM) algorithm and pseudocode for the NL-PLDA training process.

3) To verify whether there are noisy samples in the SRE04–10 dataset, we utilize the network predictions and NL-PLDA estimation to filter out high-confidence noisy samples and then verify them by human validation. As a result, we find that approximately 1% of the samples in the SRE04–10 dataset are mislabeled. After removing these samples, a relatively correct *spk2utt* mapping is released for this dataset.

This paper is organized as follows. Section II reviews the three most popular models used in speaker recognition systems. Section III introduces our proposed method for front-end learning with label noise. A detailed EM algorithm description for NL-PLDA is presented in Section IV and Appendix A. Section V shows the comprehensive experiments, results, and analyses. Section VI shows applications of the proposed method. Conclusions are given in Section VII.

## II. Embedding-Based Speaker Recognition

### A. X-Vector

Currently, deep learning-based speaker recognition is of increasing interest to researchers. The x-vector is a typical architecture that extracts discriminative low-dimensional vectors for speakers through a neural network. Benefiting from a large amount of data, the x-vector shows superior performance over the i-vector, and it is the main focus of this paper. The x-vector typically consists of frame-level layers, a pooling layer, segment-level layers, and a softmax layer. The frame-level layers process variable-length acoustic frames from speech using TDNN. The pooling layer aggregates variable-length frames into a fixed-dimensional vector. The segment-level layers are generally composed of two fully connected layers, and the outputs of the two layers are so-called x-vectors.

To deal with long-duration and variable-length training data, acoustic sequences are usually cut into multiple small chunks, and then they are used as inputs to train a network. During training, an objective function computes the cross-entropy between given speaker labels and corresponding output probabilities. In addition to the standard softmax objective function, the additive margin softmax (AM-Softmax or CosFace) [15], [45] and the additive angular margin softmax (AAM-Softmax or ArcFace) [16] are two commonly used loss functions. The backpropagation-based optimization algorithm updates the parameters of a network by minimizing the loss function. However, in the presence of noisy labels, this loss function might drive a speaker network to learn the opposite. Therefore, improving the loss function to prevent a network from fitting noisy samples is the key for learning with noisy labels.

### B. LDA

LDA is a supervised method to reduce feature dimensions, which is useful for classification tasks; therefore, it is widely used in image recognition and speaker recognition. LDA maximizes the Fisher criterion for subspace embeddings by projecting high-dimensional features into low-dimensional features through a projection matrix $\mathbf{P}$, i. e., LDA maximizes the between-class variance and minimizes the within-class variance. LDA is trained by optimizing the following Fisher criterion function:

$$\hat{\mathbf{P}} = \arg\max_{\mathbf{P}} \frac{\mathrm{tr}\left(\mathbf{P}^{\mathrm{T}}\mathbf{S}_b\mathbf{P}\right)}{\mathrm{tr}\left(\mathbf{P}^{\mathrm{T}}\mathbf{S}_w\mathbf{P}\right)}, \tag{1}$$

where $\mathbf{S}_b$ and $\mathbf{S}_w$ denote between-class and within-class variance, respectively. They are calculated as

$$\mathbf{S}_b = \frac{1}{N}\sum_{m=1}^{M} N_m \left(\boldsymbol{\mu}_m - \boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_m - \boldsymbol{\mu}\right)^{\mathrm{T}} \tag{2}$$

$$\mathbf{S}_w = \frac{1}{N}\sum_{m=1}^{M}\sum_{n=1}^{N_m} \left(\mathbf{x}_n^m - \boldsymbol{\mu}_m\right)\left(\mathbf{x}_n^m - \boldsymbol{\mu}_m\right)^{\mathrm{T}}, \tag{3}$$

where $N$ is the total number of embeddings from $M$ speakers, $N_m$ is the number of samples of the $m$-th speaker, $\boldsymbol{\mu}_m$ denotes the mean of the $m$-th speaker, $\boldsymbol{\mu}$ denotes the global mean, and $\mathbf{x}_n^m$ represents the $n$-th embedding of the $m$-th speaker. However, since LDA is a supervised model, we explore how label noise affects LDA in Appendix B.

### C. PLDA

PLDA is a probabilistic generative model typically used to make probabilistic inferences about a data class. It is a probabilistic version of LDA. Compared to LDA, PLDA adds a continuous Gaussian prior to class centers, which enables it to generate new unseen class centers given even a single example. In addition to the standard PLDA formulation [6], there are several variants of PLDA [46], such as the simplified variant [7], two-covariance variant [5], [7], and heavy-tailed PLDA [8]. In this paper, we adopt two-covariance PLDA, which is assumed to generate the class center and the observed data in a two-stage process:

$$\mathbf{y}_m \sim \mathcal{N}\left(\mathbf{y}_m|\boldsymbol{\mu}, \mathbf{B}^{-1}\right) \tag{4}$$

$$\mathbf{x}_n^m \sim \mathcal{N}\left(\mathbf{x}_n^m|\mathbf{y}_m, \mathbf{W}^{-1}\right), \tag{5}$$

where $\boldsymbol{\mu}$, $\mathbf{B}$, and $\mathbf{W}$ are the parameters estimated by PLDA, namely the global mean, between-speaker, and within-speaker precision matrices, respectively.

In the hypothesis-testing stage, given a pair of individual embeddings, we decide whether two given embeddings belong to the same speaker by computing a likelihood ratio. Although making such a decision based on cosine distance is simpler, its performance might be suboptimal under more challenging situations (e. g., cross-channel, language mismatch, and noisy environments). Due to the multiple PLDA domain adaptation technologies [37], [38], [47]–[49], and data augmentation methods [11], [50], PLDA shows its superior advantages. Under the assumption that speaker embeddings follow Gaussian distributions, PLDA has shown to be the theoretically optimal scoring method [51]. Therefore, it is currently the dominant back-end algorithm for speaker recognition.

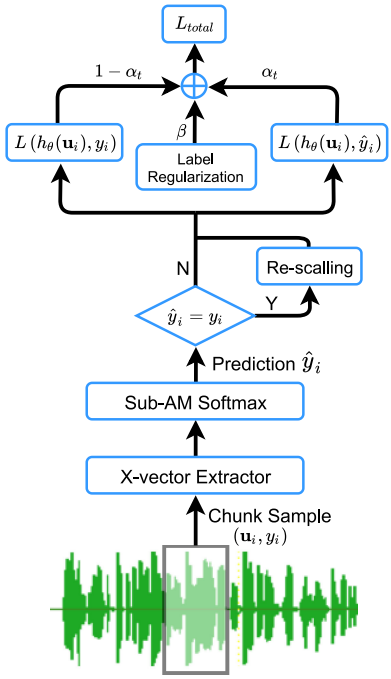



Fig. 2. An overview of the proposed method. During training, the SubAM-Softmax outputs the predicted label for each chunk, and the total loss is formed by a confidence-weighted combination of predicted and original labels.

## III. Front-End Learning With Noisy Labels

We propose a training scheme and a loss correction method to improve the front-end's label noise tolerance. As illustrated in Fig. 2, our framework consists of three major components: 1) incorporate pseudo-labels into a loss function with a well-designed dynamic confidence policy that weighs the combination of pseudo-labels and original labels; 2) rescale clean-label posterior probability; and 3) introduce a sub-center layer into AM-Softmax to separate noisy samples from training data.

### A. Label Confidence Training

To put this formally, let us first reconsider deep embedding-based speaker recognition systems from a classification perspective. The x-vector network training process is formulated as a problem of learning a model $h_\theta(\mathbf{u})$ from a set of batch training samples $\mathcal{D} = \{(\mathbf{u}_i, y_i)\}_{i=1}^{B}$, where $B$ is the mini-batch size, $y_i \in \{0,1\}^M$ denotes the ground-truth label corresponding to $\mathbf{u}_i$, and $M$ is the total number of classes. For classification issues concerning label noise, label $y_i$ might be noisy (i. e., $\mathbf{u}_i$ is a noisy sample). Supposing the extracted embedding of $\mathbf{u}_i$ is $\mathbf{x}_i$, the parameters of the network would be updated by optimizing the following loss function:

$$L = -\frac{1}{B}\sum_{i=1}^{B}\log\left(P_{i,y_i}\right), \tag{6}$$

where $P_{i,y_i}$ denotes the posterior probability that $\mathbf{x}_i$ is classified as the ground-truth label $y_i$. In this paper, we term $P_{i,y_i}$ as the *ground-truth label posterior probability*; if adopting AM-Softmax, $P_{i,y_i}$ is formulated as:

$$P_{i,y_i} = \frac{e^{s\left(\cos\left(\theta_{y_i,i}\right)-m\right)}}{e^{s\left(\cos\left(\theta_{y_i,i}\right)-m\right)} + \sum_{j\neq y_i}^{M} e^{s(\cos(\theta_{j,i}))}}, \tag{7}$$

where $m$ is the additive angular margin, $s$ is the scale factor, $\theta_{j,i}$ is the angle between $\mathbf{W}_j$ and $\mathbf{x}_i$. $\cos(\theta_j, i) = \mathbf{W}_j^{\mathrm{T}}\mathbf{x}_i$ represents the similarity score, and $\mathbf{W}_j$ is the $j$-th class center vector of the fully connected layer matrix $\mathbf{W}$. It is noteworthy that here $\mathbf{W} \in \mathbb{R}^{M\times d}$, whereas in Section III-C, the dimension is extended to $\mathbb{R}^{M\times K\times d}$ based on a sub-center layer.

However, a neural network trained directly on this objective function will overfit noisy samples, as the loss contains noisy labels. Nonetheless, prior studies have shown that deep neural networks first learn clean samples and general patterns from a dataset, and then they are forced to memorize noisy labels [21], [52]. This indicates that the network classifies noisy samples into correct classes; therefore, to leverage the network's ability to learn from data with label noise, we incorporate a *predicted label posterior probability* $P_{i,\hat{y}_i}$ into the objective function to prevent fitting incorrect samples as training iterations become larger. The subscript $\hat{y}_i \in \{0, 1, \ldots, M-1\}$ denotes the predicted label of $\mathbf{x}_i$, which is categorized as class $j$ with the max-activated output. Then, the loss function in Eq. (6) is extended as follows:

$$L' = -\frac{1}{B}\sum_{i=1}^{B}\left\{(1-\alpha_t)\log\left(P_{i,y_i}\right) + \alpha_t\log\left(P_{i,\hat{y}_i}\right)\right\}, \tag{8}$$

where $\alpha_t \in [0,1]$ is the $t$-th training iteration confidence weight between $P_{i,y_i}$ and $P_{i,\hat{y}_i}$, and it determines whether the loss function relies more on the ground-truth label or the predicted label.

The technique of Eq. (8) is similar to Bootstrapping [22]. However, Bootstrapping sets $\alpha_t$ as a fixed value for all iterations. If $\alpha_t$ is set too small, the network is influenced greatly by noisy labels during the whole training process. The resulting performance is suboptimal since the noisy label correction is limited. If $\alpha_t$ is set too large, there is a risk that the network may be too skeptical and choose labels in a random way. Therefore, a consistent balance should be established, and we adopt a dynamic weight for $\alpha_t$. Since the network parameters are initialized randomly, and the predictions are likely to be incorrect, it is not a practical idea to set $\alpha_t$ to be too large at the beginning of the training process. Additionally, $\alpha_t$ should not be set too small in an advanced stage of the training; otherwise, noisy labels will create too many adverse effects. Thus, we set $\alpha_t$ as the exponentially increasing function of the number of training iterations, formulated as:

$$\alpha_t = \alpha_T \cdot (t/T)^{\lambda}, \tag{9}$$

where $\alpha_T$ $(0 \leq \alpha_T \leq 1)$ represents the confidence weight at the final iteration $T$, $t$ denotes the number of iterations of the current training, and $\lambda$ is the exponent that controls the rate of increase. With this confidence policy, $\alpha_t$ would dynamically increase from 0 to $\alpha_T$ as the number of iterations increases. This is because the predictions become increasingly accurate during training [53], [54]. Therefore, we refer to this method as *label confidence training*.

In the case of minimizing Eq. (8), there is a risk that the network may become overconfident and predict each sample as having the same label during the last few iterations. To avoid this problem, we use mutual information $KL(P_y, P_{\hat{y}}) = P_y \log\frac{P_y}{P_{\hat{y}}}$ as label regularization [23], [55]. Label regularization measures the statistical distance between the prior probability distribution and the predicted distribution. Then, the total loss $L_{total}$ is written as:

$$L_{total} = L' + \beta \cdot P_y \log\frac{P_y}{P_{\hat{y}}}, \tag{10}$$

where $\beta$ is the regularization coefficient. To simplify the training process, we set $P_y = \frac{1}{M}$, and $P_{\hat{y}} = \frac{1}{B}\sum_{i=1}^{B} P_{i,\hat{y}}$, which is approximated by performing a calculation for each mini-batch. Label regularization allocates each sample a prior probability $P_y$ of belonging to all possible classes and helps the network prevent assigning all labels to one class.

### B. Clean Label Probability Rescaling

If the predicted label of a sample is the same as its ground-truth label, we generally believe that this sample is correctly labeled since the label confidence training criterion prevents the network from fitting noisy labels. Then, Eq. (8) is equivalent to Eq. (6). We emphasize these clean samples to utilize them to learn discriminative speaker embeddings. Intuitively, the posterior probability is larger for clean samples and smaller for noisy samples; while the loss function is a monotonically decreasing function of the posterior probability. We rescale the posterior probabilities of clean samples in the loss function, in order to increase their corresponding contribution to parameter optimization during training. Specifically, the probability for clean samples is reduced to

$$P'_{i,y_i} = \frac{e^{s\left(\cos\left(\theta_{y_i,i}\right)-m\right)}}{e^{s\left(\cos\left(\theta_{y_i,i}\right)-m\right)} + \sum_{j\neq y_i}^{M} g(u)e^{s(\cos(\theta_{j,i}))}}, \tag{11}$$

where $g(u)$ is a rescaling function, and is formulated as:

$$g(u) = e^{u \cdot s(\cos(\theta_{j,i})+1)}, \tag{12}$$

where $u \geq 0$, so that $g(u) \geq 1$, thereby $P'_{i,y_i} \leq P_{i,y_i}$ for clean samples. This idea is inspired by Co-mining [25]. However, Co-mining requires two peer networks to detect noisy samples, which makes the networks difficult to train at times. We directly reweight clean samples by the network predictions.

### C. Sub-Center AM-Softmax

AM-Softmax and AAM-Softmax are two angular margin-based objective functions that are commonly used in deep speaker recognition. The x-vectors learned by such functions are angularly distributed and naturally match the back-end scoring based on cosine similarity [56]. They also perform better than Softmax when adopting the PLDA back-end for scoring since they explicitly minimize the within-class covariance [4]. However, they are susceptible to label noise, as the inter-class speakers contain incorrect samples. To address this problem, sub-center AAM-Softmax has recently been proposed for face recognition [27], and it relaxes the intra-class constraint of AAM-Softmax [16]. In speaker recognition, AM-Softmax performs comparably to AAM-Softmax [57]–[59], and it converges faster [60]. Therefore, in this work, we adopt AM-Softmax as an objective function and relax its intra-class constraint to further improve the robustness to label noise.

The concept of "sub-classes" has been employed in face recognition for some time. Research shows that sub-classes separate different patterns more clearly, thus improving recognition accuracy [61], [62]. Following the set in [27], we introduce sub-classes into AM-Softmax and refer to the improved loss function as sub-center AM-Softmax (SubAM-Softmax for short). $K$ sub-centers are introduced in each class to relax the intra-class constraint. To form $K$ sub-centers, the dimension of matrix $\mathbf{W}$ in SubAM-Softmax is extended to $\mathbb{R}^{M\times K\times d}$, and then the similarity score is formulated as $\cos(\theta_{j,i}) = \max_k(\mathbf{W}_{jk}^{\mathrm{T}}\mathbf{x}_i), k \in \{1,\ldots,K\}$, where $\max_k$ denotes a max-pooling step. The sub-classes are able to capture the complex distribution of the training data and separate noisy samples from clean samples. Therefore, sub-classes enable the loss function to be more robust to label noise [27].

### D. Discussion

*1) Handling Hard Samples:* In this paper, we refer to hard samples as clean samples that require more time for the network to learn. In the proposed method, we relabel samples according to the network's predictions. Although the predictions are more accurate than the original noisy labels, they may also misclassify some hard samples as noisy. To reduce this mislabeling, we keep the original labels in the loss function as a part of supervised training. Moreover, in the well-designed confidence policy, as shown in Eq. (9), $\alpha_T$ controls the maximum confidence degree, and $\lambda$ controls the confidence rate for the network. Thus, the two parameters are empirically set to trade-off ground-truth and pseudo labels.

**Algorithm 1** Bayesian Estimation of PLDA with Noisy Labels.

**Input :** Traing set $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$, and corresponding labels $\mathcal{L} = \{l_1,\ldots,l_N\}$, containing $M$ individuals and $N$ samples.

Initialize $\mu$, $\mathbf{B}$, $\mathbf{W}$; $\epsilon \leftarrow 0$; $z_{n,m} \leftarrow 1$ **if** $l_n = m$ **else** $0$;

**repeat**

  **E-step:**

    Set $\mathbf{r}_y \leftarrow 0$; $\{\mathbf{R}_y^o, \mathbf{R}_y, \mathbf{R}_{xy}\} \leftarrow \mathbf{0}$;

    Update $N_e$ based on Eq. (14);

    Update $P(l_n|z_{n,m}=1,\epsilon)$ based on Eq. (27);

    **for** $m = 1$ *to* $M$ **do**

      Update $N_m$, $\mathbf{r}_{x,m}$ based on Eq. (15), (16);

      Update $\mathbf{\Phi}_m$ based on Eq. (20);

      Update $\langle \mathbf{y}_m \rangle$, $\langle \mathbf{y}_m \mathbf{y}_m^T \rangle$ based on Eq. (18), (19);

    Update $\mathbf{r}_y$, $\mathbf{R}_y^o$, $\mathbf{R}_y$, and $\mathbf{R}_{xy}$ based on Eq. (21)–(24);

  **M-step:**

    Update $\langle z_{n,m} \rangle$ based on Eq. (25);

    Update $\epsilon$ based on Eq. (28);

    Update $\mu$, $\mathbf{B}$, and $\mathbf{W}$ based on Eq. (29)–(31);

**until** *Convergence*;

**Output:** PLDA model parameters $\{\mu, \mathbf{B}, \mathbf{W}\}$, and $\epsilon$

*2) How This Method Works:* The framework leverages both the generalization ability of a network and speech signal features to learn from label noise. In generalization, a network first learns the patterns of a dataset and maintains highly robust performances at the beginning of the training process [52], which enables a model to compute correct patterns for noisy samples and classify them into correct classes before memorizing them. Therefore, the proposed method prevents a network from fitting incorrect samples by incorporating predicted labels to correct them on the fly. Additionally, a speech utterance is a one-dimensional time-series signal, and each utterance is usually divided into multiple chunks as inputs for deep neural network training [63]. When the network learned those patterns related to different speaker identities during training, it can be assumed that the network would predict correct labels for the majority of chunks in an utterance, which facilities the proposed label confidence training process.

## IV. Back-End PLDA Estimation With Noisy Labels

To address the problem of label errors in the back-end, a method for Bayesian estimation of PLDA with noisy labels is proposed in [18], [38]. The theoretical analysis of NL-PLDA has been covered extensively in [18], [38]. In this section, as a supplement, we present a detailed algorithmic presentation of NL-PLDA and its utilization of automatic filtering to weed out high-confidence noisy samples.

To combat label noise, NL-PLDA treats true labels as multinomial random variables and estimates a model's parameters based on maximum-likelihood estimation in the context of Variational Bayes. The training samples and corresponding labels are denoted as $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$, $\mathcal{L} = \{l_1,\ldots,l_N\}$, respectively, where $N$ is the sample size from $M$ individuals. However, since there are noisy samples, $\mathcal{L}$ is not the correct identity. To tackle this problem, the true label for each sample $\mathbf{x}_n$ is modeled as a latent identity $\mathbf{z}_n \in \mathbb{R}^M$, and we let $\mathcal{Z} = \{\mathbf{z}_1,\ldots,\mathbf{z}_N\}$ denote

the set of true identities corresponding to $\mathcal{X}$. The element $z_{n,m}$ ($\sum_{m=1}^{M} z_{n,m} \equiv 1$) of $\mathcal{Z}$ indicates class membership, and the expectation of $z_{n,m}$ can be seen as the confidence (probability) that $\mathbf{x}_n$ belongs to individual $m$.

The EM algorithm for the NL-PLDA is summarized in short form in Algorithm 1, and the details are available in Appendix A. In the E-step, the EM algorithm estimates both the posterior of the true identity and the individual feature distribution simultaneously; whereas the M-step updates the label error rate $\epsilon$, true latent identities $\mathcal{Z}$, and the parameters of NL-PLDA.

Moreover, since $\mathcal{Z}$ explicitly models the latent identity distribution, it can be utilized to filter out high-confidence noisy labels. Before training, no a priori information about the error rate is available; therefore, it is assumed that there are no label errors ($\epsilon = 0$). Therefore, the initialization for $\mathcal{Z}$ is shown as the left matrix of Eq. (13), where $z_{n,m}$ is initialized as $z_{n,l_n} = 1$, implying that $\mathbf{x}_n$ belongs to the original corresponding individual $l_n$. During the EM iteration steps, $\mathcal{Z}$ is determined by the maximum posterior estimation. We assume that the final updated $\mathcal{Z}$ is shown as the right matrix of Eq. (13). When the value of $z_{n,l_n}$ becomes relatively small, this indicates that $l_n$ might have a high probability of mislabeling. Therefore, a threshold (e. g., $z_{n,l_n} \leq 0.1$) is empirically set to filter out these samples.

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}, \begin{bmatrix} 0.9 & 0 & 0 & \cdots & 0.01 \\ \boxed{0.1} & 0.7 & 0.1 & \cdots & 0 \\ 0 & 0.8 & 0 & \cdots & 0.1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.4 & 0.2 & 0.2 & \cdots & \boxed{0.05} \end{bmatrix} \tag{13}$$

## V. Experiments

### A. Datasets

*1) Training Datasets:* The training datasets are prepared following SRE16 Kaldi recipe[1], including the Switchboard Phase2–3 [40], [41], Cellular1–2 (SWBD) [42], [43], and SRE04–10 [44] datasets. After filtering out nonspeech frames by energy-based voice activity detection (VAD), the recordings shorter than four seconds and the speakers with less than eight recordings are discarded. Finally, the SWBD portion contains 18,407 English recordings from 1,318 speakers, and SRE04–10 includes 2,682 speakers with 48,022 utterances. Most of these recordings are in English, while some are in Chinese, Russian, Arabic, etc. We use the two pooled datasets to train the front-end extractions, while only using the SRE04–10 portion for the LDA and PLDA back-end training.

*2) Evaluation Datasets:* The evaluation datasets consist of NIST SRE16 [64] and NIST SRE18 CMN2 [17]. Specifically, SRE16 is composed of Cantonese and Tagalog telephone conversations; the Cantonese dataset contains 965,395 trials and the Tagalog dataset contains 1,021,332 trials. For SRE18, the CMN2 collection is mainly spoken in Tunisian Arabic, and contains 108,095 trials in the development set and 2,063,007 trials in the evaluation set.

TABLE I
Configurations of the X-Vector-Based Front-Ends

| | Fixed Confidence | Dynamic Confidence | Label Regularization | Probability Rescaling | SubAM-Softmax |
|---|---|---|---|---|---|
| **Front1** | ✓ | | | | |
| **Front2** | | ✓ | | | |
| **Front3** | | ✓ | ✓ | | |
| **Front4** | | ✓ | ✓ | ✓ | |
| **Front5** | | ✓ | ✓ | | ✓ |
| **Front6** | | ✓ | ✓ | ✓ | ✓ |

"Fixed Confidence" denotes setting $\alpha_t$ in (8) to a fixed value, and "Dynamic Confidence" denotes setting $\alpha_t$ to a dynamic value.

### B. Experimental Settings

*1) Data Preparation:* In this section, we describe the way we prepare the simulated closed-set label noise. In our experiments, we first assume that the original training datasets are clean (without error labels), denoted as $\epsilon = 0\%$. To better monitor the network prediction accuracy in a clean dataset, we divide the training datasets into a *training set* and a *validation set*. The validation set is composed of one utterance randomly selected from each speaker, and the remaining recordings are used to compose the training set. It is noteworthy that there is no overlap between the two subsets. The validation set is kept clean and is used to monitor the actual prediction accuracy. To simulate different label error rates in the training set, we perform label disruption on the SWBD and SRE04–10 training sets. Specifically, we randomly select $\epsilon \in \{5\%, 10\%, 20\%, 30\%, 50\%\}$ for each speaker's utterances and then randomly relabel them as other speaker identities presented in the training set. For instance, if $\epsilon = 50\%$, then half of each speaker's utterances are randomly relabeled as belonging to other speakers in the training set to obtain a 50% label error rate in the SWBD and SRE04–10 training sets, respectively.

All of the raw audio files are converted to 40-dimensional Mel-frequency Cepstral Coefficients (MFCCs) with a 25 ms window and a 10 ms frame shift. Cepstral Mean Normalization over a three-second sliding window is applied to the MFCCs. After removing nonspeech frames by VAD, the average duration of utterances in SWBD is 171 seconds and 160 seconds in SRE04–10. For the front-end training, speech utterances are uniformly cut into chunks without overlaps, where the chunk length is set to 400 frames. These chunks are randomly formed into mini-batches as the network inputs. That is, we adopt the *sequence sampling* as designed in ASV-Subtools [65].

*2) Front-End Configurations:* We conduct experiments using x-vector-based front-ends, which are implemented in ASV-Subtools [65]. For the x-vector baseline system, we apply the extended-TDNN (E-TDNN) structure [13] with AM-Softmax loss to train a 512-dimensional x-vector extractor. The detailed implementations of the E-TDNN source codes are released on GitHub[2]. In addition to the x-vector baseline, we also train six other x-vector-based front-ends with different configurations, as shown in Table I. These front-ends are roughly trained by adding more components progressively, which enables us to observe the

[1][Online]. Available: https://github.com/kaldi-asr/kaldi/tree/master/egs/sre16/v1

[2][Online]. Available: https://github.com/Snowdar/asv-subtools/tree/master/pytorch/model

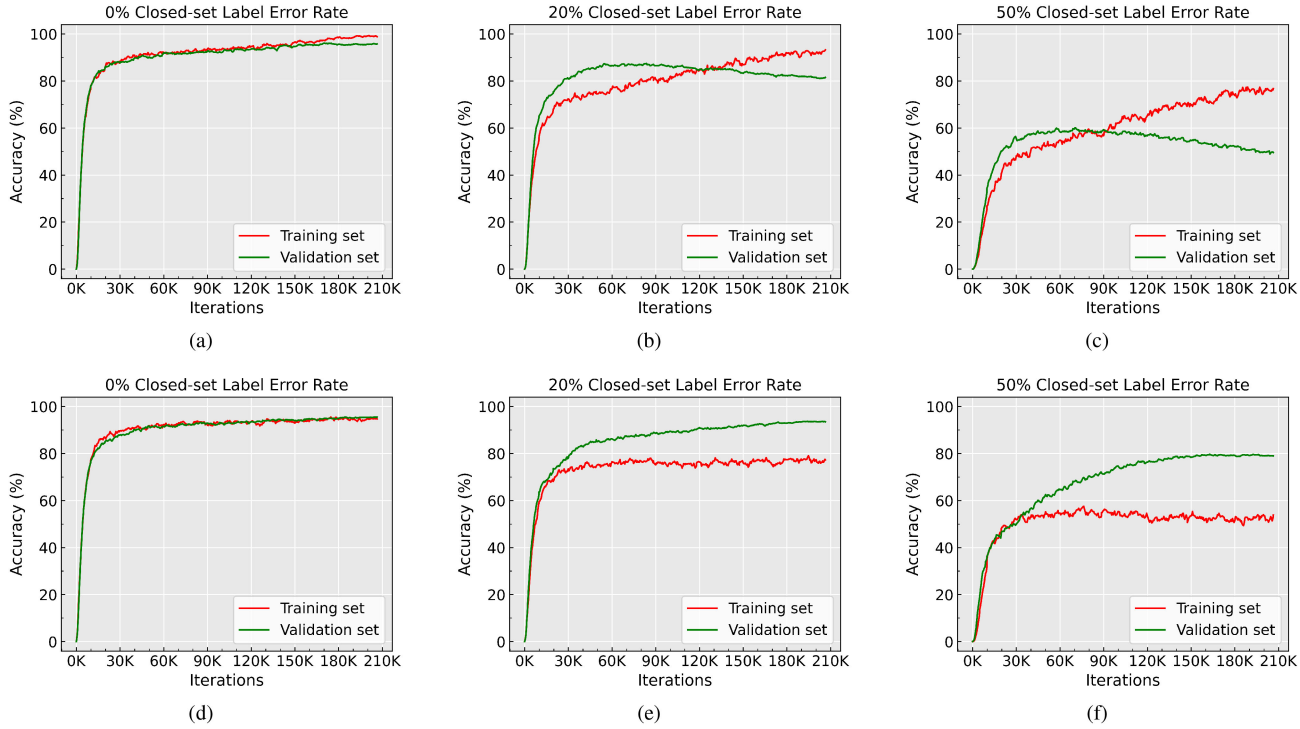



Fig. 3. Comparisons of the Baseline (Row 1) and Front2 (Row 2) accuracy under 0%, 20%, and 50% label error rates.

contribution of individual components. All of these networks are trained on GeForce RTX 2080 Ti GPUs with a mini-batch size of 256. AdamW is chosen as the optimizer, the weight decay is set to $1e-1$, and the learning rate is initially set to $1e-3$ and gradually reduced to $1e-6$. The networks are trained with 21 epochs, in which there are approximately two hundred thousand (200 K) iterations in total.

*3) Back-End Training & Evaluation:* Once trained, we choose the epoch with the highest validation set accuracy as the final model. For the evaluation process, we create one embedding per utterance for scoring purposes. The activations for the model's penultimate fully connected layer are extracted as speaker embeddings (x-vectors). Then, we project the x-vectors to a lower 256-dimensional space using LDA and adopt centering and length normalization. We trained LDA, PLDA, and NL-PLDA only on the SRE04–10 dataset. Although LDA is a supervised model, it is observed that LDA is not much affected by label noise from the explorations of LDA in Appendix B. For simplicity, we focus on the optimization of PLDA training with label noise and do not perform any label noise techniques on LDA.

Since the evaluation datasets are non-English, and PLDA is mainly trained on English utterances, domain mismatch is possible between the training and testing. To handle this problem, we adopt the unsupervised PLDA adaptation method implemented in Kaldi. For the SRE16 evaluation, the SRE16 unlabeled development set is used for PLDA adaptation. While for the SRE18 CMN2 task, the PLDA adaptation is trained on the SRE18 unlabeled set. The scoring results are reported in terms of Equal Error Rate (EER) and minimum detection cost function (minDCF) with $p$-target set to 0.01.

Experiments based on closed-set label noise are shown in Section V-C, open-set label noise experiments are shown in Section V-D, and mixed-set label noise experiments are shown in Section V-E.

### C. Validations With Closed-Set Label Noise

In this section, we compare the effects of label noise on the x-vector baseline and the Front1–6 extractors. The fixed confidence weight for Front1 is chosen as $\alpha_t = 0.3$. The exponent value for label confidence training is set to $\lambda = 2.0$, and $\alpha_T$ is set to 1.0.

*1) Training Processes:* Before presenting the final results, we first show an explicit comparison of prediction accuracy on the training set and evaluation set between the x-vector baseline and Front2. Note that the prediction accuracy is computed as the fraction of *chunk samples* in the training set or validation set that are classified correctly with respect to the corresponding labeled classes. As depicted in Fig. 3, the representative training evolutions with label error rates of $\epsilon \in \{0\%, 20\%, 50\%\}$ are presented in order from left to right. Compared with Fig. 3(a), (b), and (c), we observe that the clean dataset converges faster than the mislabeled dataset, indicating that the network takes longer to learn mislabeled samples. It also shows that the network learns reasonable representations in the first few iterations, as shown by the phenomenon that the validation set obtains higher prediction accuracy than the training set with label error rates of 20% and 50%. Unfortunately, the increasing number of training iterations does not further motivate the model to learn as expected; instead, it leads to the model overfitting incorrect samples, which subsequently degrades the prediction accuracy of the validation set.

TABLE II
PERFORMANCE COMPARISONS OF X-VECTOR BASELINE AND FRONT6 WITH CLOSED-SET LABEL ERROR RATES ($\epsilon$)

| | | PLDA | | | | | | | | NL-PLDA | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\epsilon$(%) | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | |
| | | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF |
| **Baseline** | 0 | 4.27 | 0.396 | 11.83 | 0.825 | 8.86 | 0.578 | 10.83 | 0.626 | 4.28 | 0.396 | 11.80 | 0.812 | 8.74 | 0.576 | 10.78 | 0.625 |
| | 5 | 6.41 | 0.510 | 14.15 | 0.830 | 11.84 | 0.648 | 13.66 | 0.696 | 5.72 | 0.486 | 13.49 | 0.819 | 11.20 | 0.629 | 13.05 | 0.682 |
| | 10 | 7.20 | 0.543 | 15.11 | 0.854 | 13.12 | 0.692 | 15.05 | 0.716 | 6.10 | 0.513 | 14.07 | 0.823 | 11.69 | 0.656 | 14.20 | 0.698 |
| | 20 | 9.57 | 0.620 | 17.23 | 0.898 | 14.36 | 0.761 | 17.14 | 0.768 | 7.83 | 0.585 | 15.75 | 0.846 | 12.16 | 0.714 | 15.79 | 0.747 |
| | 30 | 10.92 | 0.658 | 18.33 | 0.924 | 16.82 | 0.795 | 18.45 | 0.800 | 8.88 | 0.623 | 16.87 | 0.866 | 14.78 | 0.764 | 17.13 | 0.785 |
| | 50 | 13.70 | 0.783 | 20.65 | 0.966 | 20.65 | 0.884 | 21.41 | 0.851 | 11.99 | 0.729 | 19.70 | 0.925 | 18.94 | 0.860 | 20.53 | 0.838 |
| **Front6** | 0 | 4.01 | 0.395 | 11.57 | 0.804 | 8.44 | 0.599 | 10.59 | **0.616** | **3.99** | **0.393** | **11.53** | **0.803** | **8.31** | **0.594** | **10.55** | **0.616** |
| | 5 | 4.87 | 0.431 | 12.64 | 0.817 | 9.06 | 0.631 | 11.22 | 0.641 | **4.08** | **0.391** | **11.71** | **0.813** | **8.23** | **0.597** | **10.45** | **0.617** |
| | 10 | 5.49 | 0.470 | 13.43 | 0.808 | 9.69 | 0.640 | 11.86 | 0.658 | **4.20** | **0.408** | **11.90** | **0.801** | **8.53** | **0.602** | **10.55** | **0.621** |
| | 20 | 6.57 | 0.525 | 14.02 | 0.825 | 10.82 | 0.673 | 12.50 | 0.683 | **4.38** | **0.419** | **11.55** | **0.816** | **8.58** | **0.601** | **10.54** | **0.627** |
| | 30 | 6.84 | 0.532 | 14.67 | 0.850 | 12.31 | 0.686 | 12.94 | 0.699 | **4.36** | **0.428** | **11.70** | **0.814** | **8.90** | **0.628** | **10.54** | **0.632** |
| | 50 | 8.85 | 0.673 | 16.99 | 0.911 | 14.39 | 0.764 | 14.96 | 0.757 | **5.01** | **0.478** | **12.17** | **0.856** | **9.00** | **0.644** | **11.87** | **0.665** |

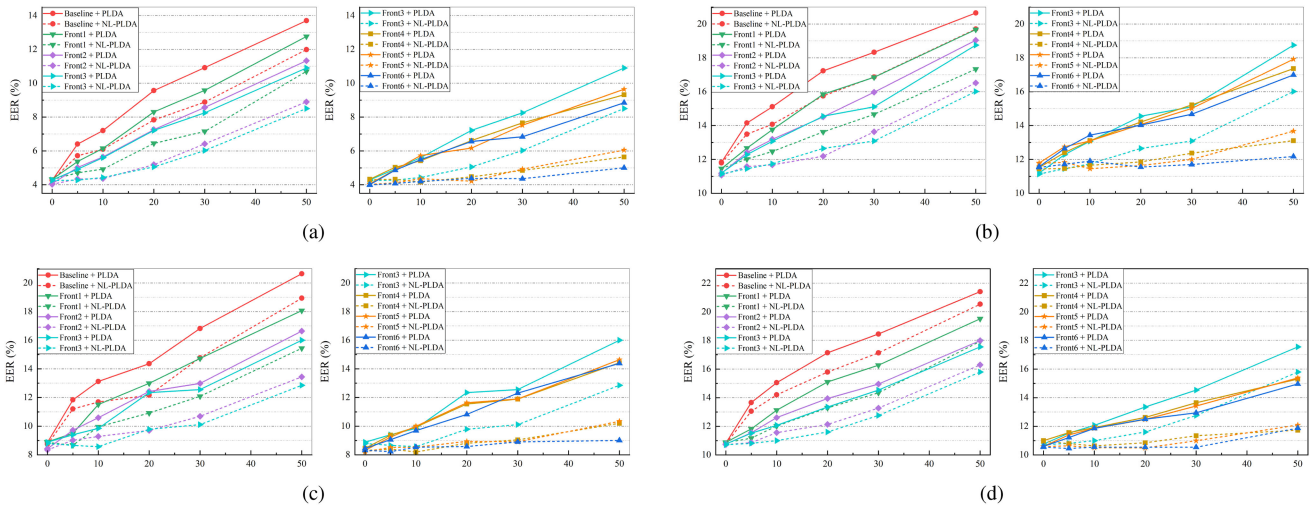



Fig. 4. Results of different front-ends used in conjunction with PLDA and NL-PLDA when encountering closed-set label errors. Results are provided for (a) the SRE16 Cantonese task, (b) the SRE16 Tagalog task, (c) the SRE18 CMN2 development task, and (d) the SRE18 CMN2 evaluation task, in terms of EER.

In contrast, the label confidence training technique suffers from fewer adverse effects due to mislabeled samples. As shown in Fig. 3(e) and (f), the model produces higher validation set accuracy during the entire training evolution. In addition, it is quite remarkable that a final validation accuracy rate of approximately 80% can be achieved even when the label error rate increases to 50%. We would like to emphasize that the final training accuracy of the models trained by this approach is close to the expected true label error rate of the dataset, indicating that this approach separates erroneous samples from correct samples within the whole training dataset.

*2) Results:* The results of the x-vector baseline and Front6 extractor used in conjunction with PLDA and NL-PLDA are summarized in Table II. To visually display the performance trends of different front-ends and back-ends, we plotted the curves for the four evaluation tasks in terms of EER, which are shown in Fig. 4.

It can be observed that the x-vector baseline performance breaks down rapidly as the label error rate increases. A 90% relative degradation in EER is obtained with a label error rate of 50%. The Front2 and Front3 extractors yield lower EERs than the baseline in situations with label noise. Additionally, compared with the results of Front1, we clearly observe that Front2 outperforms Front1 in most situations, indicating that the gradually increasing confidence weight is more effective in reducing the impact of label noise. Furthermore, when incorporating label regularization, Front3 leads to lower performance degradation relative to Front2. Therefore, in the next front-end experiments, we adopt the dynamic confidence policy with label regularization.

To examine the effectiveness of the rescaling strategy and SubAM-Softmax for handling label noise, Fig. 4 provides the results of the Front4–6 extractors, which add the rescaling, SubAM-Softmax, and the combined techniques, respectively. Compared to the results of Front3, one can observe that Front4 exhibits greater resistance to the impact of label noise and boosts the performance, which shows that adding a rescaling technique to focus more on clean labels is helpful for learning with noisy labels. Additionally, comparisons of the results of Front3 and Front5 show that SubAM-Softmax is more robust than the standard AM-Softmax in dealing with label noise.

Next, we observe the experimental results of NL-PLDA, which are shown by the dashed lines in Fig. 4. For the different front-ends, the NL-PLDA results are consistent with the trend of PLDA as the label noise rates increase; while NL-PLDA achieves better performance in terms of EER and minDCF

TABLE III
PERFORMANCE COMPARISONS OF THE BASELINE AND FRONT6 WITH OPEN-SET LABEL ERROR RATES ($p$)

| | | PLDA | | | | | | | | NL-PLDA | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $p$(%) | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | |
| | | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF |
| **Baseline** | 5 | 4.91 | 0.440 | 12.27 | 0.833 | 9.16 | 0.632 | 11.24 | 0.655 | 4.53 | 0.424 | 11.89 | 0.824 | 8.67 | 0.617 | 10.82 | 0.644 |
| | 10 | 5.67 | 0.476 | 12.51 | 0.834 | 10.11 | 0.675 | 11.92 | 0.673 | 5.10 | 0.453 | 11.98 | 0.821 | 9.31 | 0.654 | 11.30 | 0.658 |
| | 20 | 6.01 | 0.488 | 13.41 | 0.840 | 10.51 | 0.676 | 12.23 | 0.685 | 5.30 | **0.462** | 12.48 | 0.822 | 9.59 | 0.659 | 11.49 | 0.675 |
| | 30 | 7.06 | 0.539 | 14.60 | 0.861 | 11.25 | 0.698 | 13.27 | 0.707 | 6.61 | **0.523** | 13.36 | **0.836** | 10.41 | 0.705 | 12.44 | 0.704 |
| | 50 | 8.75 | 0.637 | 16.49 | 0.914 | 13.30 | 0.719 | 14.52 | 0.730 | 7.86 | 0.569 | **14.07** | **0.852** | 12.15 | 0.759 | 13.70 | **0.718** |
| **Front6** | 5 | 4.66 | 0.425 | 11.74 | 0.795 | 8.54 | 0.599 | 10.67 | 0.638 | **4.29** | **0.408** | **11.27** | **0.786** | **7.93** | **0.576** | **10.24** | **0.626** |
| | 10 | 5.39 | 0.454 | 12.16 | 0.817 | 9.02 | 0.627 | 11.19 | 0.646 | **4.72** | **0.437** | **11.52** | **0.801** | **8.39** | **0.616** | **10.61** | **0.632** |
| | 20 | 5.95 | 0.493 | 13.08 | 0.825 | 9.81 | 0.657 | 12.03 | 0.671 | **5.13** | 0.474 | **12.01** | **0.811** | **8.70** | **0.654** | **11.31** | **0.660** |
| | 30 | 6.80 | 0.535 | 14.02 | 0.864 | 11.10 | 0.680 | 12.67 | 0.687 | **6.07** | 0.524 | **13.03** | 0.850 | **9.92** | **0.680** | **11.91** | **0.682** |
| | 50 | 8.08 | 0.594 | 15.95 | 0.894 | 11.71 | 0.756 | 14.14 | 0.746 | **7.32** | **0.561** | 14.94 | 0.859 | **10.34** | **0.739** | **13.03** | 0.726 |

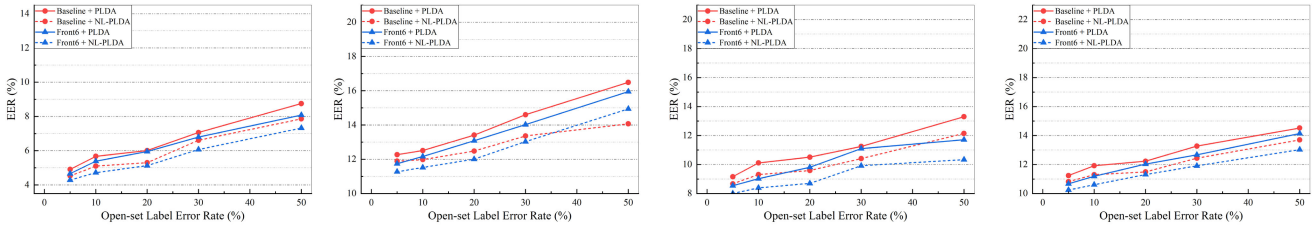



Fig. 5. Performance comparisons of the Baseline and Front6 used in conjunction with PLDA and NL-PLDA in the situation of open-set label noise. Results are provided for (a) the SRE16 Cantonese task, (b) the SRE16 Tagalog task, (c) the SRE18 CMN2 development task, and (d) the SRE18 CMN2 evaluation task, in terms of EER.

than simply using PLDA scoring. This implies that NL-PLDA is capable of handling label noise. However, the performance of NL-PLDA degrades in the presence of strong label noise, especially for the x-vector. One explanation is that an insufficient number of correct labels poses a challenge to NL-PLDA training. Another important reason is that the speaker embeddings learned by the x-vector contain less discriminative information, which confuses the NL-PLDA label noise estimation. This implies that a front-end that is robust to label noise would facilitate the whole system's performance improvement. Additionally, when Front6 is used in conjunction with the NL-PLDA back-end, it yields significant performance improvements relative to the baseline methods when the training datasets are noisy. Due to the space limitation, we demonstrate the detailed results of Front1–Front5 in the website[3], as well as the minDCF curves.

From the results presented above, we draw the following conclusions: 1) Substantial improvements are obtained by label confidence training. 2) Rescaling and SubAM-Softmax have complementary properties, and the systems yield further improvements when used in tandem. 3) NL-PLDA always performs better than PLDA when training labels are noisy. 4) The superiority of NL-PLDA benefits from a more robust front-end, especially in the presence of strong label error rates.

### D. Validations With Open-Set Label Noise

In this section, we conduct experiments to validate the performance of the improved front-ends and NL-PLDA with open-set noisy datasets. The open-set noisy datasets are simulated by randomly selecting $p \in \{5\%, 10\%, 20\%, 30\%, 50\%\}$ training utterances per speaker in the original SWBD and SRE04–10 datasets and replacing them with utterances randomly selected from the concatenated VoxCeleb2 datasets. (Subsegments belonging to the same video are concatenated together to form a unique utterance, and then it is downsampled to 8 kHz). The labels and the number of utterances per speaker remain unchanged. We use open-set noisy datasets to train the x-vector baseline and Front6 extractors.

Experimental results for four evaluation datasets with PLDA and NL-PLDA scoring are shown in Table III, the EER curves are shown in Fig. 5. As shown, Front6 achieves better performances compared to the x-vector baseline, which indicates that the proposed method robustly trains front-ends from open-set noisy datasets. Although these noisy samples cannot be localized to the corresponding correct labels in the training set, it can be assumed that this method reduces their detrimental effects by assigning them to similar classes from the training set.

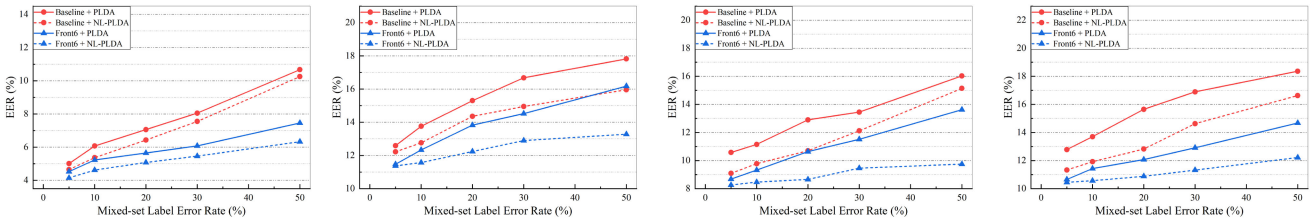
### E. Validations With Mixed-Set Label Noise

In this section, we consider the mixed condition with both closed-set label noise and open-set label noise. The mixed-set label noise rate is denoted by $\tau \in \{5\%, 10\%, 20\%, 30\%, 50\%\}$. The proportions of open-set noise and closed-set noise in the mixed-set are both 50%. The preparations for closed-set and open-set noisy samples are the same as those in Section V-B and Section V-D. The results are presented in Table IV, and the EER curves for the four evaluation datasets are shown in Fig. 6. We clearly see that Front6 consistently outperforms the x-vector baseline with mixed-set label noise. In addition, the NL-PLDA effectively models mixed label noise and is more robust. In summary, synthetic experiments reveal that our method is powerful enough to handle mixed label noise, particularly in cases of high label noise rates.

[3][Online]. Available: http://dwz.date/fbC5

TABLE IV
PERFORMANCE COMPARISONS OF THE X-VECTOR BASELINE AND FRONT6 WITH MIXED-SET LABEL ERROR RATES ($\tau$)

| | | PLDA | | | | | | | | NL-PLDA | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\tau$(%) | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | | SRE16 Cantonese | | SRE16 Tagalog | | SRE18 CMN2 Dev | | SRE18 CMN2 Eval | |
| | | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF |
| **Baseline** | 5 | 5.01 | 0.453 | 12.59 | 0.831 | 10.58 | 0.651 | 12.78 | 0.687 | 4.65 | **0.409** | 12.22 | 0.826 | 9.09 | 0.624 | 11.33 | 0.652 |
| | 10 | 6.07 | 0.489 | 13.76 | 0.843 | 11.15 | 0.697 | 13.70 | 0.690 | 5.37 | 0.476 | 12.76 | 0.831 | 9.78 | 0.650 | 11.92 | 0.665 |
| | 20 | 7.06 | 0.503 | 15.30 | 0.854 | 12.90 | 0.704 | 15.65 | 0.734 | 6.43 | 0.537 | 14.36 | 0.847 | 10.70 | 0.742 | 12.82 | 0.710 |
| | 30 | 8.05 | 0.598 | 16.68 | 0.882 | 13.45 | 0.743 | 16.89 | 0.767 | 7.55 | 0.548 | 14.95 | 0.856 | 12.12 | 0.743 | 14.63 | 0.749 |
| | 50 | 10.67 | 0.702 | 17.82 | 0.895 | 16.03 | 0.845 | 18.36 | 0.862 | 10.25 | 0.627 | 15.96 | 0.877 | 15.14 | 0.785 | 16.63 | 0.751 |
| **Front6** | 5 | 4.53 | 0.412 | 11.47 | **0.762** | 8.68 | 0.607 | 10.66 | 0.622 | **4.15** | 0.417 | **11.37** | 0.776 | **8.26** | **0.598** | **10.45** | **0.621** |
| | 10 | 5.23 | 0.438 | 12.33 | 0.768 | 9.32 | 0.648 | 11.43 | 0.656 | **4.62** | **0.437** | **11.57** | **0.781** | **8.47** | **0.618** | **10.57** | **0.634** |
| | 20 | 5.65 | 0.442 | 13.83 | 0.825 | 10.64 | 0.719 | 12.07 | 0.657 | **5.08** | **0.443** | **12.24** | **0.809** | **8.66** | **0.645** | **10.88** | **0.641** |
| | 30 | 6.08 | 0.488 | 14.52 | 0.865 | 11.51 | 0.721 | 12.91 | 0.697 | **5.46** | **0.487** | **12.90** | **0.838** | **9.46** | **0.651** | **11.32** | **0.651** |
| | 50 | 7.46 | 0.573 | 16.18 | 0.899 | 13.62 | 0.745 | 14.68 | 0.755 | **6.33** | **0.491** | **13.28** | **0.854** | **9.75** | **0.733** | **12.22** | **0.695** |




(a) (b) (c) (d)

Fig. 6. Performance comparisons of the Baseline and Front6 used in conjunction with PLDA and NL-PLDA in the situation of mixed-set label noise. Results are provided for (a) the SRE16 Cantonese task, (b) the SRE16 Tagalog task, (c) the SRE18 CMN2 development task, and (d) the SRE18 CMN2 evaluation task, in terms of EER.

## VI. APPLICATIONS

### A. Label Correction for Synthetic Datasets

As shown above, when a model is trained to be robust to noisy labels, the prediction accuracy of the model is greater than the correct label rate of the data. One straightforward application is to relabel a full utterance using its chunk-level predictions from a trained model, and we call this application *label correction*. This is of practical interest, as label-corrected datasets can then be beneficially used to retrain front-end networks and back-end models.

To verify this, we apply label correction to synthetic closed-set, open-set, and mixed-set label noise. The Front6 extractors trained on different label error rates are utilized to predict the chunk-level labels for the corresponding dataset. In the label-prediction process, the inputs for the network are the chunk-level samples from each utterance, while the output is the speaker label corresponding to each chunk sample. Then, each utterance is relabeled with the predicted label that occurs with more than 50% frequency in its multiple chunks, which is seen as the dominant label. If an utterance does not have a dominant label, this utterance would be discarded, as it is likely to be an outlier.

The experimental results show that the utterance-level prediction accuracy of a dataset with label noise is further improved through label correction, and the updated label error rate is significantly reduced compared to the original error rate. For example, the closed-set error rate is reduced from: 5% $\rightarrow$ 1.2%, 10% $\rightarrow$ 1.4%, 20% $\rightarrow$ 1.9%, 30% $\rightarrow$ 2.6%, and 50% $\rightarrow$ 8.6%, which implies that this method corrects erroneous labels even in cases with high error rates.

We then use the label-corrected datasets to retrain the speaker recognition systems. The comparisons of Front6 trained with and without label correction for the SRE18 CMN2 evaluation task are shown in Fig. 7. We clearly observe that great improvements have been achieved for all types of label noise, namely closed-set, open-set, and mixed-set. These improvements show that label correction alleviates adverse effects from mislabeling in all scenarios.

### B. Label Denoising for Real-Word SRE04–10 Dataset

We further investigate whether there are mislabels in the original "clean" SRE04–10 dataset. This experiment adopts more rigorous methods, including network prediction, NL-PLDA estimation, and human validation; we call this process *label denoising*. After being trained on the original dataset, Front6 and NL-PLDA are used to filter out high-confidence noisy samples. Two types of samples are filtered out: those with predicted labels that are inconsistent with ground-truth labels and those with low latent identities ($z_{n,m} < 0.1$). We obtain a subdataset that is 1.2% the size of the original SRE04–10 dataset. Then, we identify these erroneous labels by human validation; we find that more than half of these samples are indeed mislabeled. For example, some speaker utterances are identified as being spoken by a male when they are in fact spoken by a female and vice versa. Furthermore, some utterances contain ambient sound in which a speaker's voice can barely be heard. Examples of mislabeled audio files are publicly available[4]. Finally, we retrain the x-vector baseline on the original dataset with these samples removed, and the results are shown in Table V. Compared with the results of the x-vector trained without the use of label denoising, slight improvements are observed in the SRE18 CMN2 development

[4][Online]. Available: http://dwz.date/fbC5

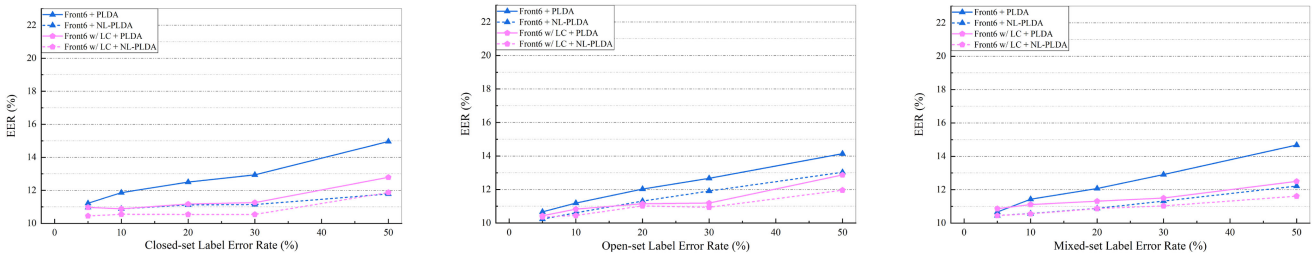



Fig. 7. Performance comparisons of Front6 trained with and without the label correction datasets, (a) label correction in the situation of closed-set label noise, (b) label correction in the situation of closed-set label noise, (c) label correction in the situation of mixed-set label noise. Results are provided for the SRE18 CMN2 evaluation task in terms of EER, where "LC" denotes label correction.

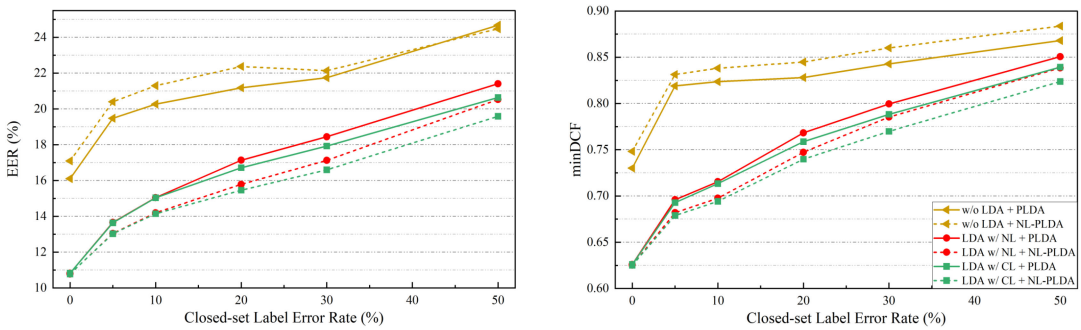



Fig. 8. Performance comparisons of different LDA configurations, where "NL" denotes noisy labels and "CL" denotes clean labels. The SRE18 CMN2 evaluation results of the x-vector baseline with PLDA and NL-PLDA in situation of closed-set label noise are provided in terms of EER and minDCF.

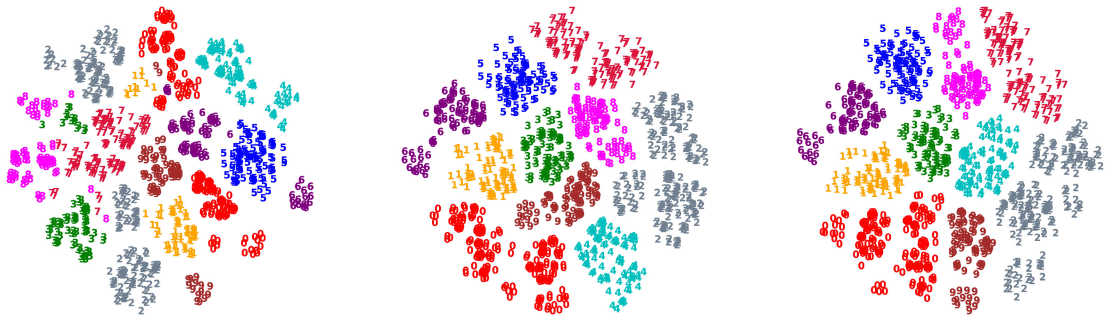



Fig. 9. t-SNE visualization of speaker embeddings without LDA, LDA trained on noisy labels, and LDA trained on clean labels (in order from left to right, respectively). The speaker embeddings are extracted by the x-vector baseline trained with a 50% label error rate. Each number or color represents a class.

TABLE V
RESULTS OF X-VECTOR TRAINED WITH AND WITHOUT LABEL DENOISING FOR THE SRE18 CMN2 TASK

| SRE18 CMN2 | Label Denoising | PLDA | | NL-PLDA | |
|---|---|---|---|---|---|
| | | EER | minDCF | EER | minDCF |
| Dev | No | 8.86 | 0.578 | 8.74 | 0.576 |
| | Yes | 8.59 | 0.576 | **8.56** | **0.576** |
| Eval | No | 10.83 | 0.626 | 10.78 | 0.625 |
| | Yes | **10.55** | 0.622 | 10.56 | **0.621** |

and evaluation test sets. In addition, a relatively clean version of the SRE04–10 *spk2utt* file that contains speaker-to-utterance mappings is also available in the website[4].

## VII. CONCLUSION

In this paper, we demonstrate that label noise leads to significant performance degradation in both the x-vector front-end and PLDA back-end. Then, we propose a simple yet effective approach to combat label noise during front-end training. Our proposed framework contains three strategies, including a label confidence training scheme, a posterior probability rescaling strategy, and an improved AM-Softmax loss function. When progressively combining these three strategies, experiments conducted on the pooled SWBD and SRE04–10 datasets show consistent improvements in robustness against label noise. Since a speaker recognition system consists of both a front-end and a back-end, it is necessary to optimize both to achieve the

best performance. Consequently, we also optimize the back-end PLDA when the training labels are noisy. When combining the optimized front-end and back-end, the whole speaker recognition system demonstrates strong resistance to noisy labels.

In addition, we show two practical applications for this improved system, including label correction and label denoising. Label correction is used to correct noisy labels that occur in a dataset. We propose correcting noisy samples based on utterance chunk-level predictions from a well-trained network. The experimental results show that label correction greatly reduces the number of noisy samples. Therefore, models retrained on a label-corrected dataset perform similarly to those trained on a clean dataset. We apply label denoising to the original SRE04–10 dataset to weed out the original erroneous labels, and both the front-end and back-end are used to algorithmically filter out high-confidence noisy samples. Then, we verify the sample labels with human validation. Through verification, approximately 1% of the samples are found to be noisy in the original SRE04–10 dataset. Experimental results show that models trained on the label-denoised datasets achieve slight improvements compared to the baseline system.

In the future, we are interested in validating our method on other front-end networks and conducting experiments on real-world datasets with noisy labels. In addition, we plan to apply this approach to semi-supervised learning and self-supervised learning networks.

## Appendix A
## PLDA Learning Algorithm With Noisy Labels

This Appendix presents the EM algorithm for the parameter $\{\boldsymbol{\mu}, \mathbf{B}, \mathbf{W}\}$ learning in NL-PLDA [18]. For convenience, let $\langle\cdot\rangle$ denote the expectation of a given random variable.

In the E-step, let us first pre-compute the number of erroneous samples as:

$$N_e = \sum_{n=1}^{N} \left(1 - \langle z_{n,l_n} \rangle\right), \tag{14}$$

the number of samples for the $m$-th individual:

$$N_m = \sum_{n=1}^{N} \langle z_{n,m} \rangle, \quad 1 \le m \le M, \tag{15}$$

the first-order moment for the $m$-th individual:

$$\mathbf{r}_{x,m} = \sum_{n=1}^{N} \langle z_{n,m} \rangle \mathbf{x}_n, \quad 1 \le m \le M, \tag{16}$$

and the global second-order moment:

$$\mathbf{R}_x = \sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}}. \tag{17}$$

Then, we compute the first and second moments of the latent variables:

$$\langle \mathbf{y}_m \rangle = \boldsymbol{\Phi}_m^{-1} \left(\mathbf{B}\boldsymbol{\mu} + \mathbf{W}\mathbf{r}_{x,m}\right), \tag{18}$$

$$\left\langle \mathbf{y}_m \mathbf{y}_m^{\mathrm{T}} \right\rangle = \boldsymbol{\Phi}_m^{-1} + \langle \mathbf{y}_m \rangle \langle \mathbf{y}_m \rangle^{\mathrm{T}}, \tag{19}$$

where

$$\boldsymbol{\Phi}_m = \mathbf{B} + N_m \mathbf{W}. \tag{20}$$

Next, we need to compute the following auxiliary matrices:

$$\mathbf{r}_y = \sum_{m=1}^{M} \langle \mathbf{y}_m \rangle, \tag{21}$$

$$\mathbf{R}_y^o = \sum_{m=1}^{M} \left\langle \mathbf{y}_m \mathbf{y}_m^{\mathrm{T}} \right\rangle, \tag{22}$$

$$\mathbf{R}_y = \sum_{m=1}^{M} \sum_{n=1}^{N} \langle z_{n,m} \rangle \left\langle \mathbf{y}_m \mathbf{y}_m^{\mathrm{T}} \right\rangle, \tag{23}$$

$$\mathbf{R}_{xy} = \sum_{m=1}^{M} \mathbf{r}_{x,m} \langle \mathbf{y}_m \rangle^{\mathrm{T}}. \tag{24}$$

For the M-step, we update the matrix of label latent identity $\mathcal{Z}$ by:

$$\langle z_{n,m} \rangle = \frac{i_{n,m}}{\sum_{j=1}^{M} i_{n,j}}, \tag{25}$$

where

$$i_{n,m} = P\left(l_n \mid z_{n,m} = 1, \epsilon\right) \mathcal{N}\left(\mathbf{x}_n \mid \langle \mathbf{y}_m \rangle, \mathbf{W}^{-1}\right) \times \exp\left(-\frac{1}{2} \operatorname{tr}\left\{\mathbf{W}\boldsymbol{\Phi}_m^{-1}\right\}\right) \tag{26}$$

$$P\left(l_n \mid z_{n,m} = 1, \epsilon\right) = \begin{cases} 1 - \epsilon, & \text{if } l_n = m \\ \frac{\epsilon}{M-1}, & \text{else} \end{cases} \tag{27}$$

$$\epsilon = \frac{N_e}{N}. \tag{28}$$

After that we update the NL-PLDA parameters as follows:

$$\boldsymbol{\mu} = \frac{\mathbf{r}_y}{M}, \tag{29}$$

$$\mathbf{B} = M \left(\mathbf{R}_y^o - \frac{\mathbf{r}_y \mathbf{r}_y^{\mathrm{T}}}{M}\right)^{-1}, \tag{30}$$

$$\mathbf{W} = N \left(\mathbf{R}_x - \mathbf{R}_{xy} - \mathbf{R}_{xy}^{\mathrm{T}} + \mathbf{R}_y\right)^{-1}. \tag{31}$$

## Appendix B
## Effects of LDA Configurations

Since LDA training also requires speaker labels, in this set of experiments, we investigate the effects of different LDA configurations on PLDA and NL-PLDA. Three distinct back-end configurations are compared concretely: without LDA, LDA trained on noisy labels, and LDA trained on clean labels, respectively. The x-vector baseline is used as the front-end, and the experimental results are shown in Fig. 8. From the results, it is clear that the back-end without LDA yields the worst results; moreover, NL-PLDA trained without LDA loses its ability to combat label noise and even achieves worse results than PLDA trained without LDA. However, the performance of NL-PLDA is improved by using LDA projection, even if the LDA is trained with noisy labels. LDA trained on clean labels achieves optimal results. By comparing the conditions of LDA with clean labels

and with noisy labels, we found that the relative performance gap is within 5% of the EER values with NL-PLDA in the back-end, even under a label error rate of 50%.

To further observe the effects of LDA, we visualize the speaker embeddings by plotting the t-SNE embeddings. The embeddings from ten distinct clusters with different LDA configurations are shown in Fig. 9. Each embedding is represented by its corresponding true label. From Fig. 9(a), it is apparent that embeddings without LDA projections are more isolated within their classes, and they do not have clearly separated boundaries between classes. While in Fig. 9(b) and (c), the embeddings with LDA are more effectively separated into clusters. The embeddings presented in Fig. 9(b) are very similar to those in Fig. 9(c), demonstrating that LDA shows insensitivity to label noise. It also suggests that LDA removes non-discriminant dimensions, which are potentially caused by label noise, and LDA facilitates PLDA and NL-PLDA training.

## REFERENCES


[1] K. A. Lee *et al.*, "NEC-TT system for mixed-bandwidth and multi-domain speaker recognition," *Comput. Speech Lang.*, vol. 61, 2020, Art. no. 101033.

[2] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Process Lett*, vol. 13, no. 5, pp. 308–311, May 2006.

[3] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011.

[4] J. Villalba *et al.*, "State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and speakers in the wild evaluations," *Comput. Speech Lang.*, vol. 60, 2020, Art. no. 101026.

[5] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 531–542.

[6] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[7] N. Brümmer and E. De Villiers, "The speaker partitioning problem," in *Proc. Odyssey*, 2010, pp. 194–201.

[8] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Proc. Odyssey*, 2010, p. 14.

[9] Y. Cai, L. Li, A. Abel, X. Zhu, and D. Wang, "Deep normalization for speaker vectors," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 733–744, Dec. 2021.

[10] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, 2011, pp. 249–252.

[11] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5329–5333.

[12] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 3214–3218.

[13] D. Snyder *et al.*, "Speaker recognition for multi-speaker conversations using x-vectors," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5796–5800.

[14] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 1–5.

[15] H. Wang *et al.*, "CosFace: Large margin cosine loss for deep face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5265–5274.

[16] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4690–4699.

[17] S. O. Sadjadi *et al.*, "The 2018 NIST speaker recognition evaluation," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 1483–1487.

[18] B. J. Borgström and P. Torres-Carrasquillo, "Bayesian estimation of PLDA with noisy training labels, with applications to speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7594–7598.

[19] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Comput. Speech Lang.*, vol. 60, 2020, Art. no. 101027.

[20] R. Sachdeva, F. R. Cordeiro, V. Belagiannis, I. Reid, and G. Carneiro, "Evidentialmix: Learning with combined open-set and closed-set noisy labels," in *Proc. Winter Conf. Appl. Comput. Vis.*, 2021, pp. 3607–3615.

[21] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[22] S. E. Reed *et al.*, "Training deep neural networks on noisy labels with bootstrapping," 2014, *arXiv:1412.6596*.

[23] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5552–5560.

[24] E. Arazo, D. Ortego, P. Albert, N. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 312–321.

[25] X. Wang, S. Wang, J. Wang, H. Shi, and T. Mei, "Co-mining: Deep face recognition with noisy labels," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9358–9367.

[26] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," 2020, *arXiv:2007.08199*.

[27] J. Deng, J. Guo, T. Liu, M. Gong, and S. Zafeiriou, "Sub-center ArcFace: Boosting face recognition by large-scale noisy web faces," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 741–757.

[28] J. Yao *et al.*, "Deep learning from noisy image labels with quality embedding," *IEEE Trans Image Process*, vol. 28, no. 4, pp. 1909–1922, Apr. 2019.

[29] S. Jenni and P. Favaro, "Deep bilevel learning," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 618–633.

[30] R. Tanno, A. Saeedi, S. Sankaranarayanan, D. C. Alexander, and N. Silberman, "Learning from noisy labels by regularized estimation of annotator confusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11 244–11 253.

[31] D. Hendrycks, K. Lee, and M. Mazeika, "Using pre-training can improve model robustness and uncertainty," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2712–2721.

[32] Y. Lyu and I. W. Tsang, "Curriculum loss: Robust learning and generalization against label corruption," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–22.

[33] T. Nguyen, C. Mummadi, T. Ngo, L. Beggel, and T. Brox, "Self: Learning to filter noisy labels with self-ensembling," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–16.

[34] S. Zheng, G. Liu, H. Suo, and Y. Lei, "Towards a fault-tolerant speaker verification system: A regularization approach to reduce the condition number," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 4065–4069.

[35] M. Pham, Z. Li, and J. Whitehill, "How does label noise affect the quality of speaker embeddings," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 3216–3220.

[36] X. Qin, N. Li, C. Weng, D. Su, and M. Li, "Simple attention module based speaker verification with iterative noisy label detection," 2021, *arXiv:2110.06534* .

[37] B. J. Borgström, "Unsupervised bayesian adaptation of PLDA for speaker verification," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 1039–1043.

[38] B. J. Borgström, "Bayesian estimation of PLDA in the presence of noisy training labels, with applications to speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 414–428, 2022.

[39] F. Tong, Y. Liu, J. Wang, L. Li, and Q. Hong, "Automatic error correction for speaker embedding learning with noisy labels," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 4628–4632.

[40] D. Graff, K. Walker, and A. Canavan, "Switchboard-2 phase II," 1999. Accessed: Apr. 17, 2022. [Online]. Available: https://catalog.ldc.upenn.edu/LDC99S79

[41] D. Graff, D. Miller, and K. Walker, "Switchboard-2 phase III," 2002. Accessed: Apr. 17, 2022. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2002S06

[42] D. Graff and K. Walker, "Switchboard cellular part 1 audio," 2001. Accessed: Apr. 17, 2022. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2001S13

[43] D. Graff and K. Walker, "Switchboard cellular part 2 audio," 2004. Accessed: Apr. 17, 2022. [Online]. Availabel: https://catalog.ldc.upenn.edu/LDC2004S07
[44] S. O. Sadjadi, "NIST SRE CTS superset: A large-scale dataset for telephony speaker recognition," 2021, *arXiv:2108.07118*.
[45] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Process. Lett.*, vol. 25, no. 7, pp. 926–930, Jul. 2018.
[46] A. Sizov, K. A. Lee, and T. Kinnunen, "Unifying probabilistic linear discriminant analysis variants in biometric authentication," in *Proc. Joint IAPR Int. Workshops Stat. Techn. Pattern Recognit. (SPR) Struct. Syntactic Pattern Recognit. (SSPR)*, 2014, pp. 464–475.
[47] D. Garcia-Romero, A. McCree, S. Shum, N. Brummer, and C. Vaquero, "Unsupervised domain adaptation for i-vector speaker recognition," in *Proc. Odyssey: Speaker Lang. Recognit. Workshop*, 2014, pp. 260–264.
[48] Y. Tu, M.-W. Mak, and J.-T. Chien, "Variational domain adversarial learning with mutual information maximization for speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2013–2024, 2020.
[49] K. A. Lee, Q. Wang, and T. Koshinaka, "The CORAL algorithm for unsupervised domain adaptation of PLDA," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5821–5825.
[50] S. Wang, Y. Yang, Z. Wu, Y. Qian, and K. Yu, "Data augmentation using deep generative models for embedding based speaker recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2598–2609, 2020.
[51] D. Wang, "Remarks on optimal scores for speaker recognition," 2020, *arXiv:2010.04862*.
[52] D. Arpit *et al.*, "A closer look at memorization in deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 233–242.
[53] X. Liu, S. Li, M. Kan, S. Shan, and X. Chen, "Self-error-correcting convolutional neural network for learning with noisy labels," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit.*, 2017, pp. 111–117.
[54] Y. Wang *et al.*, "Iterative learning with open-set noisy labels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8688–8696.
[55] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama, "Learning discrete representations via information maximizing self-augmented training," in *Proc. Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1558–1567.
[56] Z. Bai and X. Zhang, "Speaker recognition based on deep learning: An overview," *Neural Netw.*, vol. 140, pp. 65–99, 2021.
[57] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2019, pp. 1652–1656.
[58] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 2873–2877.
[59] Z. Chen, Z. Ren, and S. Xu, "A study on angular based embedding learning for text-independent speaker verification," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2019, pp. 445–449.
[60] M. Rybicka and K. Kowalczyk, "On parameter adaptation in softmax-based cross-entropy loss for improved convergence speed and accuracy in DNN-based speaker recognition," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 3805–3809.
[61] M. Zhu and A. M. Martínez, "Optimal subclass discovery for discriminant analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2004, pp. 97–97.
[62] H. Wan, H. Wang, G. Guo, and X. Wei, "Separability-oriented subclass discriminant analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 2, pp. 409–422, Feb. 2018.
[63] A. K. Sarkar, Z.-H. Tan, H. Tang, S. Shon, and J. Glass, "Time-contrastive learning based deep bottleneck features for text-dependent speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 8, pp. 1267–1279, Aug. 2019.
[64] S. O. Sadjadi *et al.*, "The 2016 NIST speaker recognition evaluation," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 1353–1357.
[65] F. Tong *et al.*, "ASV-Subtools: Open source toolkit for automatic speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 6184–6188.

**Lin Li** (Member, IEEE) received the B.S. and Ph.D. degrees in electronic engineering from the University of Science and Technology of China, Hefei, China, in 2003 and 2008, respectively. From 2008 to 2011, she was an Assistant Professor with Electronic Engineering Department, Xiamen University, Xiamen, China. Since 2011, she has been an Associate Professor. She is the Vice Director of the Department of Electronic Engineering, Xiamen University. Her research interests include speaker/language recognition, speech signal processing, and applications. She is a China Computer Federation (CCF) Member and a Committee Member of the National Conference on Man-Machine Speech Communication.

**Fuchuan Tong** received the B.S. degree in electronic and information engineering from Sichuan University, Chengdu, China, in 2019. He is currently working toward the master's degree with XMU Speech Lab, School of Electronic Science and Engineering, Xiamen University, Xiamen, China. His current research interests mainly include deep learning based speaker recognition and speaker diarization.

**Qingyang Hong** (Member, IEEE) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in January 2005. After that, he began his job with Xiamen University, Xiamen, China. He is currently an Associate Professor with the School of Informatics. He has authored or coauthored more than 60 papers and one book. His research interests include speaker recognition, speech recognition, and applications of deep learning.