applied sciences

MDPI

Article

U-Vectors: Generating Clusterable Speaker Embedding from Unlabeled Data

Muhammad Firoz Mridha [1], Abu Quwsar Ohi [1], Muhammad Mostafa Monowar [2], Md. Abdul Hamid [2], Md. Rashedul Islam [3,*] and Yutaka Watanobe [4]

1 Department of Computer Science & Engineering, Bangladesh University of Business & Technology, Dhaka 1216, Bangladesh; firoz@bubt.edu.bd (M.F.M.); quwsarohi@gmail.com (A.Q.O.)
2 Department of Information Technology, Faculty of Computing & Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; mmonowar@kau.edu.sa (M.M.M.); mabdulhamid1@kau.edu.sa (M.A.H.)
3 Department of Computer Science and Engineering, University of Asia Pacific, Dhaka 1216, Bangladesh
4 Department of Computer Science and Engineering, University of Aizu, Aizu-Wakamatsu 965-8580, Japan; yutaka@u-aizu.ac.jp
* Correspondence: rashed.cse@gmail.com

Abstract: Speaker recognition deals with recognizing speakers by their speech. Most speaker recognition systems are built upon two stages, the first stage extracts low dimensional correlation embeddings from speech, and the second performs the classification task. The robustness of a speaker recognition system mainly depends on the extraction process of speech embeddings, which are primarily pre-trained on a large-scale dataset. As the embedding systems are pre-trained, the performance of speaker recognition models greatly depends on domain adaptation policy, which may reduce if trained using inadequate data. This paper introduces a speaker recognition strategy dealing with unlabeled data, which generates clusterable embedding vectors from small fixed-size speech frames. The unsupervised training strategy involves an assumption that a small speech segment should include a single speaker. Depending on such a belief, a pairwise constraint is constructed with noise augmentation policies, used to train AutoEmbedder architecture that generates speaker embeddings. Without relying on domain adaption policy, the process unsupervisely produces clusterable speaker embeddings, termed unsupervised vectors (u-vectors). The evaluation is concluded in two popular speaker recognition datasets for English language, TIMIT, and LibriSpeech. Also, a Bengali dataset is included to illustrate the diversity of the domain shifts for speaker recognition systems. Finally, we conclude that the proposed approach achieves satisfactory performance using pairwise architectures.

Keywords: speaker recognition; clustering; twin networks; deep learning

check for updates

Citation: Mridha, M.F.; Ohi, A.Q.; Monowar, M.M.; Hamid, M.A.; Islam, M.R.; Watanobe, Y. U-Vectors: Generating Clusterable Speaker Embedding from Unlabeled Data. *Appl. Sci.* **2021**, *11*, 10079. https://doi.org/10.3390/app112110079

Academic Editor: Byung-Gyu Kim

Received: 1 October 2021
Accepted: 22 October 2021
Published: 27 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Speech is the most engaging and acceptable form of communication among one another. Artificial intelligence (AI) systems are currently continuously targeting and working on various challenges of speech-related topics, including speech recognition, speech segmentation, speaker recognition, speech diarization, and so on. Among the different sub-domains of AI, Deep learning (DL) strategies often perform superior to other techniques.

The general implementation of DL was mainly conducted on speech recognition systems. DL methods can be trained on speech recognition without the requirement of speech-to-word alignment [1]. Often such training strategies are defined as an end-to-end method. End-to-end methods can be easily trained from speech transcripts. Therefore, currently, numerous systems are being developed in the speech recognition domain. Speech recognition systems have exciting usages in voice commands, virtual assistants, search

engines, speech-to-text processing, etc. Further, numerous automation systems on speech are currently being developed. A speech denoising mechanism removes environmental sounds from speech audio, helping to provide a clean speech [2]. Speech synthesis systems artificially enable computers to produce speech sounds [3], helping computers to communicate with humans. Speech emotion systems further extract human emotions from speech [4]. Thus, computers can apprehend human feelings. Speech segmentation systems can segment a speech into word/phone levels [5], helping to identify words and phones from speech. Further, computers can help humans in developing pronunciation [6]. Among the various speech-based solutions, speaker recognition has fascinating usage of identifying users by hearing speech.

Speaker recognition systems are directly involved with biometric identification systems and are suitable for authenticating users remotely by hearing voices . In perspective to various biometric systems, such as facial recognition, fingerprint matching, and so on, speaker recognition also has vast usability in numerous domains, including telecom, banking, search optimization, and diarization [7]. Nevertheless, speaker recognition systems suffer difficulties, including speech states, emotional conditions, environmental noise, health conditions, speaking styles, etc. Further, in comparison with supervised speaker recognition approaches, unsupervised and semi-supervised strategies are hardly investigated [8]. Unsupervised and semi-supervised systems resolve the requirement of labeling a vast quantity of speech data.

DL architectures have been extensively investigated for supervised speaker recognition systems. For speaker and speech recognition models, speech spectrograms and mel-frequency cepstral coefficients (MFCC) [9] are used as a preprocessing strategy. For such cases, convolutional neural networks (CNN) are generally implemented [10]. However, current architectures processes raw-audio and extract speaker recognizable features. SincNet [11] improves the feature extraction process from raw audio waves. The architecture fuses sinc functions with CNN that can extract speaker information from low and high cutoff frequencies. AM-MobileNet1D [12] further demonstrates that 1D-CNN architectures are sufficient for identifying features from raw audio waveforms. Also, the architecture requires fewer parameters compared to SincNet. Although supervised speaker recognition architectures perform excellently in recognition tasks on a large set of speakers. However, DL strategies require a vast amount of labeled data to operate on speech-related queries.

Generating speech embeddings has been widely observed in the speaker recognition domain [13–15]. Embedding refers to generating vectors of continuous values. Often, architectures using speaker embeddings are also termed stage-wise architectures [7]. Currently, unsupervised speaker recognition systems implement domain adaptation policies, mostly fused with embedding vectors [7,16]. Domain adaptation refers to finding appropriate similarities, where a framework is trained on training data, and tested on a similar yet unseen data. Hence, domain adaptation strategies may perform poorly when the variation of training and the unseen dataset is massive. Further, domain adaptation strategies may also produce less accuracy if trained on an inadequate dataset with minimal variation [15]. Yet, efforts have been made to reduce the interpretation of unseen data over training data, by adding adversarial training [17], improving training policies [18], covariance adaptation policies [19], etc. Although these policies improve the robustness, most strategies are still prone to various speech diversions, such as language, speech pattern, age, emotion, and so on.

Currently, in the aspect of DL, embeddings can be generated using triplet [20] and pairwise loss [21] techniques. In triplet loss architecture, three parallel inputs flow through the network: anchor, negative, and positive. The positive input contains a similar class w.r.t. to the anchor, whereas the negative input contains a different class. Comparatively, in pairwise architecture, a pair of information flows either belonging to a single or different class. Triplet architectures have been perceived in speaker feature extraction in supervised practice [22].

This paper introduces an unsupervised strategy of generating speaker embedding directly from the unseen data. Hence, the method does not depend on domain adaptation policies and can adapt diverse features from most speech data. Moreover, we insist on converting DL architecture's training process to both semi-supervised and unsupervised manners . Yet, to do so, the system requires segmented audio streams (length of 1 s) and needs to guarantee that a segment contains only one person's speech. The audio segment is further windowed into smaller speech frames (0.2 s) for training the DL architecture. The audio segments are assigned pseudo labels, which are further reconstructed by DL architecture. Figure 1 illustrates the construction of the training procedure.

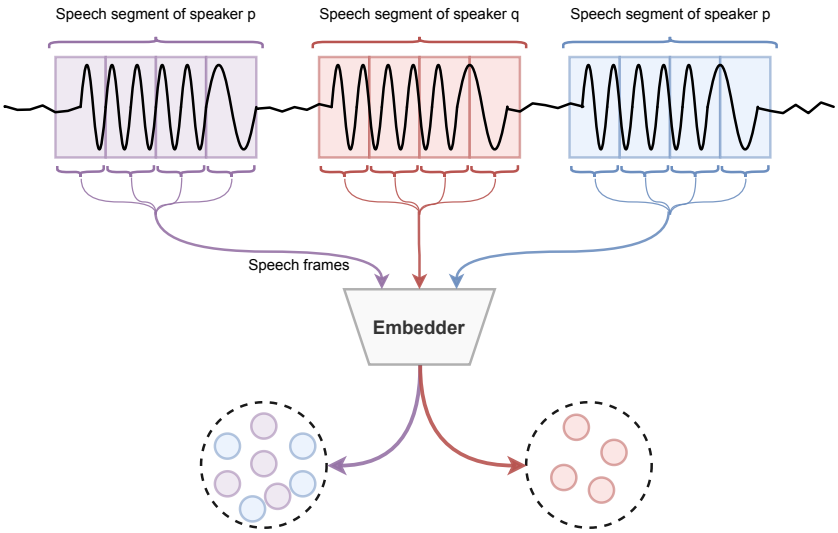


Figure 1. The figure illustrates a set of segmented speech with an unknown number of speakers (in the example, two speakers, p and q). Speech segments are windowed into smaller speech frames, assuming that all frames of a single speech segment belong to a single class. Further, a DL-based embedding system finds speech similarities (inter-segment similarity) and relations from speech segments. The process results in generating clusterable speaker embeddings.

The overall contributions of the paper are the following:

- We introduce a strategy of generating speaker-dependent embeddings, named u-vector. The training process is domain-independent and directly learns from the unlabeled data.
- We use pairwise constraints and non-generative augmentations to train AutoEmbedder architecture.
- We explore the possibilities of our strategy in both unsupervised and semi-supervised training criteria.
- We evaluate the proposed policy with two inter-cluster based strategies: triplet and pairwise architectures.
- We finally conclude that a DL architecture can discriminate speakers from pseudo labels based on feature similarity.

We organize the paper as follows, Section 2 reviews the related works conducted in speaker recognition domain. Section 3 clarifies the construction of the training procedure, along with the challenges, and modifications. Section 4 illustrates the experimental setup, datasets and the analysis of the architecture's performance. In Section 5, we sketch the proposed method's future initiatives along with usabilities. Finally, Section 6 concludes the paper.

2. Related Work

Speaker recognition has been a topic of interest over the past decades, and various systems have been proposed to solve the challenge. In the domain of speaker recognition,

numerous techniques have been observed since late 2000. Among these, embedding architectures have been widely explored to extract the diversity of speech frames. Embedding models are often considered feature extractors, which can generate a speech-print (related to finger-print) of an individual. Hence, every individual's speech will remain closer in the embedding space, causing to create a cluster of embeddings from speech frames.

Gaussian mixture model (GMM) supervector [23] (stacking mean vectors from GMM), and Joint factor analysis (JFA) [24] have been popularly integrated into the speaker recognition task. JFA merges the speaker-dependent and channel-dependent supervector and generates a new supervector based on the dependency. GMM and JFA were significantly accepted as feature extractors and implemented in various speaker recognition strategies. Later on, inspired by JFA, identity vector (i-vector) [25] was introduced. I-vector contributes to changing the channel-dependent supervectors and integrates speaker information within the supervectors. Hence, i-vector became more sensitive to speech variations and greatly accepted by the researchers. In most cases of JFA and i-vector, MFCC is widely implemented. MFCC is a linear cosine transform of a log power spectrum used to extract a sound's information. However, a lower MFCC with a lower cepstral coefficient returns only sound information, whereas the higher value of the coefficient represents speaker information as well [26]. Further, probabilistic linear discriminant analysis (PLDA) [27] is mostly used for implementing speaker verification and identification systems using i-vectors [13].

The present improvement of DL architectures has led to revisiting the speech embedding representation neural architecture perspective. Deep vector (d-vector) [14] is a mutated implementation of the speech frame embeddings using deep neural networks (DNN). The d-vector depends on the automated feature extraction process of DNN. The model's training process is supervised, and in the basic implementation of the d-vector, it is explored as a text-dependent system. After the training procedure, the softmax layer is left out, and the embeddings are extracted from the last hidden layer. Although the d-vector is based on DNN, further studies have been made using CNN architectures [28]. In the modified architecture, speech is converted into MEL coefficients, which are normalized and supplied to CNN. Moreover, extensive studies have been made to improve the basic d-vector to a text-independent unsupervised vector generation using domain adaptation [29]. The mechanism is split into two parts in the upgraded version: a DNN that extracts embeddings and a separately trained classifier that classifies speakers. These studies' limitation is that most of them require massive labelled data in the training procedure. Also, the embedding performance in the case of unseen speakers dramatically depends on the training data.

As DNN architectures are dependent on the amount of training data, an improved strategy of the d-vector is proposed, named x-vector [15]. X-vectors are a modified version of d-vector, which depends on basic sound augmentation techniques, noise and reverberation. Further, the implementation highly motivates data augmentation usage and presents a decent accuracy improvement over i-vectors. The default x-vector is implemented based on the improved text-independent version of the d-vector [29] by properly utilizing data augmentation.

The present state of the art speech embedding systems tends to be unsupervised. However, the concept of unsupervision still depends on a large set of training data. Both d-vector and x-vectors directly rely on the domain adaptation [30] policy of neural network architectures. Hence, the performance of these architectures on unseen data massively depends on the volume and diversity of the training dataset. The domain adaptation capability of neural network architectures is further increased by using synthetic datasets [31]. However, the performance is still dependent on previously learned features, and performance might lack due to data inefficiency and domain variation between training and testing data.

Therefore, we introduce an approach that is independent of domain adaptation of neural network architectures. Instead, the proposed method tends to utilize the automated feature extraction of neural network.

3. Methodology

This paper's clusterable speaker embedding generation is based on a particular assumption: a speaker speaks continuously for a specific time. Hence, if segmentation methods are used or a small speech segment is extracted based on voice segmentation techniques, most speech segments will contain an individual's speech. However, some segments might be impure, i.e., a single segment may have multiple individuals' speech. Nevertheless, we argue that the ratio of impurity would be small enough for most general speech conversations. Hence, such a strategy is investigated with the most common neural network pipelines; a siamese network [32]. AutoEmbedder framework [21] is used as a DL architecture to extract speaker embeddings. Figure 2 illustrates the basic workflow of producing u-vectors.

In this section, the methodology of generating u-vector is introduced. The section is segmented as follows: the problem formulation and assumptions are defined in Section 3.1. The proposed work includes speech segments discussed in Section 3.2. Further, speech segments are broken into small speech frames for construction pairwise constraint, which is addressed in Section 3.3. Uncertainties due to the pseudo-labels of pairwise constraints are discussed in Section 3.4. Challenges of deciding segmentation length are addressed in Section 3.5. Finally, the DL framework AutoEmbedder [21], which is used to explore the actual cluster linkage, is theorized in Section 3.6.

3.1. Problem Formulation and Assumptions

The proposed method tends to solve the speaker recognition system in an unsupervised manner based on some constraints. Table 1 summarizes the paper's mathematical notations to facilitate the readers. To comprehend the problem statement, let $\mathcal{S}$ be a database of speech segment, where $\mathcal{X}_k$ be a short-length audio segment containing speech of an individual. Also, let x_i be a smaller window/frame of the audio segment, where $x_i \in \mathcal{X}_k$. From a particular speech segment, $\mathcal{X}_k$, $\mathcal{M}$ number of non-overlapping speech frames are generated. As it is stated that a speech segment belongs to a single individual, the smaller speech frames also belong to that individual. From this intuition, we construct pairwise constraints between audio frames. We define two speech frames belong to the same cluster if they belong to the same audio segment. On the contrary, we consider two speech frames that belong to different clusters if they belong to different audio segments. Based on the pairwise relations, a set of cluster $\mathcal{C}$ can be generated. Where a single cluster ($c_i \in \mathcal{C}$) belongs to a specific speech segment. Considering most speech segments will belong to a single speaker, we can assume that most cluster c_i would contain a single individual's data. However, as multiple speech segments can belong to a single individual, multiple clusters may contain a single individual's data. Hence, the challenge is to find such optimal cluster relationships such that no two clusters may contain speech of a single individual. The training strategy has two possibilities, such as:

- In the case of an unlabeled dataset, let us consider that each audio file contains a single individual's speech. For a set of audio files with an unknown number of speakers, our approach is suitable to produce clusterable embeddings based on speakers. This constraint is similar to semi-supervised learning, as some of the pairwise constraints are known [33].
- Let us consider a dataset containing multiple speakers' conversations, where a single audio stream may include various speakers. In such a case, no pairwise constraints are known, and an unsupervised strategy is required. Hence, we produce hypothetical pairwise constraints based on audio segmentation processes such as VAD, word segmentation [5], etc., and construct pairwise constraints. However, in

such a case, the embedding system's accuracy depends on the purity of the audio segmentation process.

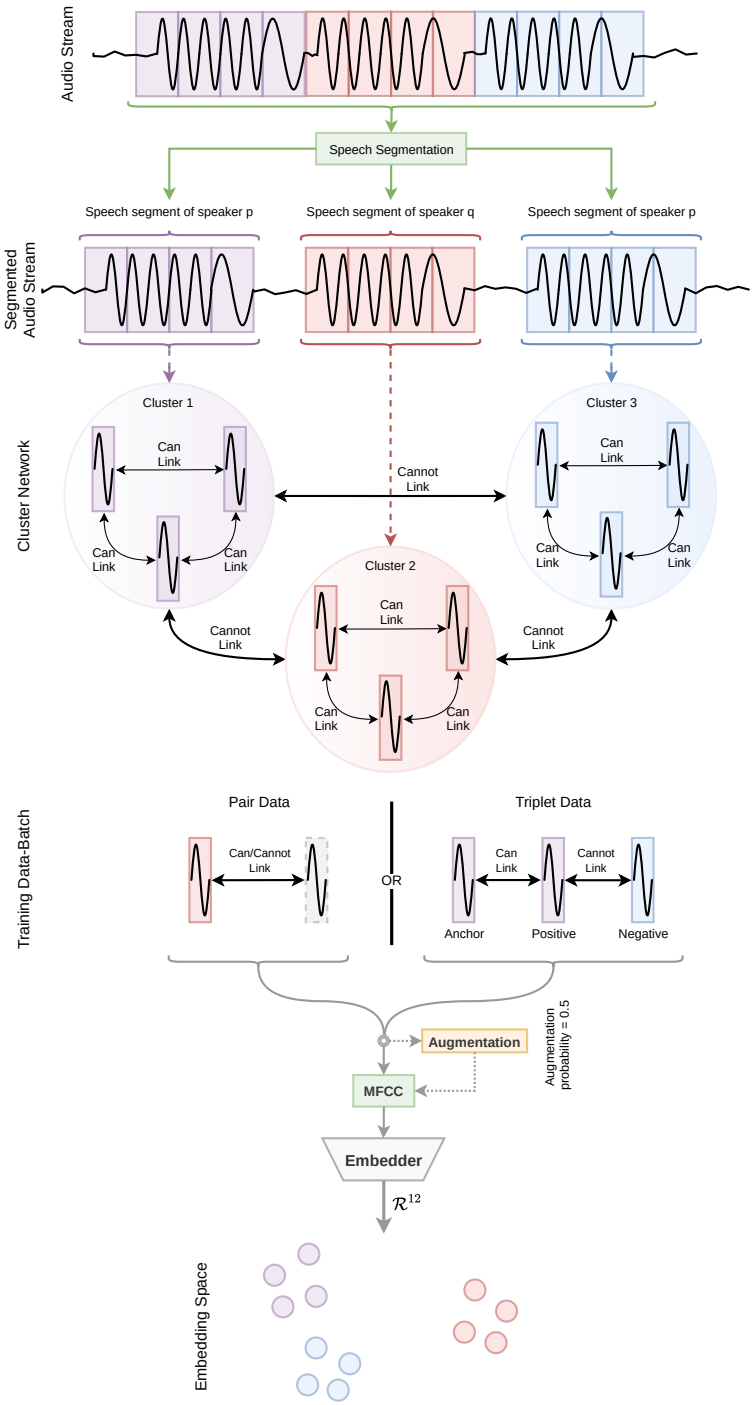


Figure 2. The figure illustrates the comprehensive procedure to generate u-vectors. Audio stream is segmented and given pseudo labels. Then a cluster network is formed using small non-overlapping speech frames from the speech segments. Finally, training data is generated from the cluster network based on the needs of the siamese architecture. The pairwise network requires an equal number of can-link and cannot link pairs. In contrast, triplet networks receive three data, one pair with can-link, and one cannot-link pair. The network/embedder is trained on randomly augmented speech frames, further converted into 2D feature maps using MFCC. While training, the embedder reforms the cluster associations, where speaker similarity is considered.

For both of the problem, DL architecture is used to aggregate multiple clusters such that the resulting cluster contains all of the embeddings of a single individual. We imply that if a DL function can properly extract speech features from audio frames, it can obtain optimal reasoning of speech frames being similar and dissimilar. Further, an optimally trained DL framework can successfully re-cluster the data based on the feature similarity rather than the number of hypothetical clusters. We extend our investigation towards finding such DL training strategy.

Since the approach deals with unsupervisely generating speaker embedding vectors from speech data (without domain adaptation), the output embedding vectors are named as unsupervised vectors (u-vectors). Formally u-vector is defined as follows,

Definition 1. *Unsupervised vectors (u-vectors) refer to a set of DNN generated speech embeddings, which are clusterable based on speakers, trained from unlabeled speech segments.*

In the formal definition, by DNN, any specific implementation of substantially deep neural networks are indicated, such as convolutional, feedforward, recurrent, etc.

Table 1. The mathematical notations used in the paper are summarized.

Notation	Description
$\mathcal{S}$	A set of audio segments. Audio segments are fragments of a continuous audio stream. We assume that most audio segments contain speech of a single individual.
$\mathcal{X}$	A single audio segment, $\mathcal{X} \in \mathcal{S}$.
x_i	An audio frame, generated by taking shorter frames from an audio segment, $x_i \in \mathcal{X}_k$. Audio frames are used to train DL architectures.
$\mathcal{M}$	Denotes the number of possible audio frames in an audio segment, $x_{1 \leq i \leq \mathcal{M}} \in \mathcal{X}_k$. Theoretically, $\mathcal{M} \times \lvert x_i \rvert = \lvert \mathcal{X}_k \rvert$.
$\mathcal{C}$	A set of clusters. These clusters are formed using hypothetical pairwise constraints. As cluster linkages are constructed based on the speech segment relations, it can be considered that $\lvert \mathcal{S} \rvert = \lvert \mathcal{C} \rvert$.
c_i	Denotes a subset of the entire cluster, $c_i \subseteq \mathcal{C}$. Here, c_i represents a cluster constructed using the inter-relationship of speech frames, belonging to a specific speech segment $\mathcal{X}_i$.
$\mathcal{N}$	The actual number of individuals in $\mathcal{S}$, considering the ground truth. For this specific problem, the value of $\mathcal{N}$ is unknown.
α	The distance hyperparameter used for AutoEmbedder [21] architecture. For other architectures, α may indicate a connectivity state for any two cluster nodes.

3.2. Speech Segments

To generate the pairwise constraints, it is considered that a speech segment belongs to an individual. Moreover, if it is possible to extract accurate pairwise constraints, a DL framework can be trained using those constraints. To generate such pairwise constraints, speech segmentation procedures are required.

Speech can be easily segmented using various techniques. Methods such as VAD [34] and word segmentation [35] can be indeed adopted to define such speech segments, containing the voice of a single individual. It is also feasible to assume that a single individual mostly speaks more than one word in a conversation. Hence, it is also possible to queue multiple speech segments and hypothesize that they come from a single individual. However, increasing the queue or size of a speech segment also increases the probability of impurity of a speech segment (discussed in Section 3.5). By impurity in a speech segment, it is referred that a speech segment contains more than one speaker. Impure data can often trick the DL frameworks from finding actual relationships among clusters. Hence, to minimize the impurity risk, we study speech segments with a length of one second. After the successful extraction of speech segments, the pairwise constraints are to be constructed. Although the overall framework is dependent on proper speech segmentation techniques, we avoid implementing such segmentation methods. Instead, we provide a detailed evaluation of embedding accuracy based on various levels of cluster impurities.

3.3. Pairwise Constraints

The DL framework is trained based on pairwise constraints. Pairwise constraint contains a pairwise relationship between a pair of inputs. By considering x_i and x_j as two random speech frames, two circumstances may occur: (a) speech frames may belong to the same audio segment $\mathcal{X}_k$ or (b) they may belong to different audio segments. In the current state of the problem, as the speech labels' ground truth is unknown for every speech segment, we consider each segment belonging to different individuals. Hence, the number of unique pseudo labels is equal to $|\mathcal{S}|$. Mathematically, $\mathcal{C}$ being a set of clusters, c_i being a particular cluster of similar nodes, and $\mathcal{X}_k$ being a specific speech segment,

$$\begin{aligned} \forall x_i \in \mathcal{X}_k \;\; and \;\; \forall x_j \in \mathcal{X}_k, \quad x_i, x_j \in c_k \\ \forall x_i \in \mathcal{X}_k \;\; and \;\; \forall x_j \notin \mathcal{X}_k, \quad x_i, x_j \notin c_k \end{aligned} \tag{1}$$

The DL framework is trained based on the defined cluster constraints. To properly introduce the inter-cluster and intra-cluster relation to a DL framework, we define a gound regression function based on pairwise criteria derived in Equatoin (1). The function is defined as,

$$\mathcal{P}_c(x_i, x_j) = \begin{cases} 0 & \text{if } x_i, x_j \in c_p \\ \alpha & \text{if } x_i \in c_p \text{ and } x_j \in c_q \end{cases} \tag{2}$$

In general, the $\mathcal{P}_c(\cdot, \cdot)$ outputs the distance constraints that each embedding (generated from speech frames) holds. The function infers that an embedding pair must be at a close distance if they belong to the same cluster or at a distance of α otherwise. However, embedding pairs belonging to different clusters may be at a distance greater than α, which is established in the AutoEmbedder architecture (Equatoin (4)). We use the pairwise constraints to train a DL architecture. Further, we revisit the data-clusters' uncertainty and segmenting impurities and explore why a DL framework may be necessary in such a case.

3.4. Uncertainties in Pairwise Constraints

The cluster assignments are mostly uncertain based on two major concerns: (a) the segmented audio $\mathcal{X}_k$ may be impure, (b) the ground-truth of cluster assignments are unknown. Therefore, in most cases, the number of ground-label (defined as $\mathcal{N}$) is theoretically not equal to the number of clusters, i.e., $\mathcal{N} \neq |\mathcal{C}|$ and $\mathcal{N} \neq |\mathcal{S}|$, where $|\mathcal{S}| = |\mathcal{C}|$. Moreover, due to such impurity and uncertainty of ground-labels, the subsequent flaws in the training dataset (based on pairwise properties) are frequently observed,

- Impurity in must-link constraint: The dataset's core concept is to assume that an audio segment $\mathcal{X}_k$ contains only one individual's speech. Generally, a segmentation system may inaccurately identify speech segments and hold multiple individuals' speech in a single audio segment. However, if we consider short length audio segments, the probability of speaker fusion rapidly decreases.
- Error in cannot-link constraint: Let, $x_i \in c_p$ and $x_j \in c_q$, where $c_p \neq c_q$. The cluster assignments are considered based on the number of audio segments. Hence, for most datasets, the number of speech segments is greater than the actual number of speakers, $|\mathcal{C}| \geq \mathcal{N}$. Therefore, considering the ground-truth, the assumption $c_p \neq c_q$ may be wrong, and data pair x_i and x_j may belong to the same cluster considering the ground truth.

If we consider a cluster network $\mathcal{C}$ with no impurity, then the task of DL is to eliminate the errors in cannot link constraints based on the feature relationship. Hence, if it is possible to prioritize the speech features to a DL framework, it can allegedly aggregate appropriate cluster from erroneous cannot-link clusters. Therefore, training the DL architecture reduces errors in cannot-link constraints. However, reducing the impurity of the input data's must-link constraints considerably depends on the length of speech segments and segmentation policies.

3.5. Segment Length Analysis

The time-domain length of the speech segments (defined by $|\mathcal{X}|$) operates a vital role in the overall performance of the training process. Each segment is further windowed into smaller speech frames. Hence, the segment length must be divisible by the length of fixed-size speech frames (defined by $|x|$). Various architectures consider overlapped frames while windowing speech signals. However, we avoid such measures, as such overlaps result in mixing similar speech patterns in multiple speech frames.

To illustrate the trade-off of selecting an optimal length of speech segment $|\mathcal{X}|$, let us consider $\mathcal{L}_{mean}$ being the mean and $\mathcal{L}_{std}$ the standard deviation (std) of the length of speech segmentations for a given dataset (or a buffer of audio stream). Therefore, statistically, $\mathcal{L}_{mean} - \mathcal{L}_{std}$ is the optimal minimal length for which we can assume that most segments strictly contains speech of a single individual. However, if the minimum segment length is considered, the number of frames per segment $\mathcal{M}$ would also reduce.

Reducing the number of frames per segment due to a shorter segment would deliver less inter-cluster relations for each segment. The reduction of inter-cluster association would also cause the DL framework struggle finding feature relation between speech frames. Further, increasing the size of speech segments may also result in impure components, if $|\mathcal{X}| \geq \mathcal{L}_{mean} - \mathcal{L}_{std}$.

To explore the reason of impurity, let us consider an audio stream contains a mean time $\mathcal{J}_{mean}$ with standard deviation of $\mathcal{J}_{std}$, after which, the speaker exchanges. In such condition, selecting the length of segment too high may result in being $|\mathcal{X}| \geq \mathcal{J}_{mean} - \mathcal{J}_{std}$. However, statistically, in most general conversations, the length of minimal speech segmentation is mostly less than the speaker exchange time, $\mathcal{L}_{mean} + \mathcal{L}_{std} \leq \mathcal{J}_{mean} - \mathcal{J}_{std}$. Therefore, if we can select such $|X|$, for which, $|\mathcal{X}| < \mathcal{L}_{mean} - \mathcal{L}_{std} < \mathcal{J}_{mean} - \mathcal{J}_{std}$, the rate of impurity would be zero. Hence, selecting $|\mathcal{X}| \approx \mathcal{L}_{mean}$ would reduce the rate of impurity. For the experimental datasets, the $\mathcal{L}_{mean}$ is equal to one (illustrated in Table 2). Hence, we experiment with one-second speech segment. Further, we investigate a pairwise framework in which we try to trick DL architecture into converging towards the ground cluster relationship.

Table 2. The table illustrates the mean, median and deviation of segment duration and words per sentence for each dataset. The segment duration is calculated using the setup described in Section 4.1.

Dataset	Segment Duration			Words per Sentence		
	Mean	Median	STD	Mean	Median	STD
TIMIT [36]	1	0.8	0.6	8.63	8.0	2.6
LibriSpeech [37]	1.2	0.8	1	18.9	15.0	12.9
Bengali ASR [38]	1.3	1.2	0.8	3.20	3.0	3.0

3.6. AutoEmbedder Architecture

As a DL architecture, we use a pairwise constraint-based AutoEmbedder framework to re-cluster speech data. However, we introduce further modifications to the network's general training process to strengthen the learning progress. In general, the AutoEmbedder architecture is trained based on the pairwise constraints defined by function $\mathcal{P}_c(\cdot,\cdot)$. The architecture follows siamese network constraints that can be presented as,

$$\mathcal{S}(x, x^{'}) = ReLU(\|\mathcal{E}_\phi(x) - \mathcal{E}_\phi(x^{'})\|, \alpha) = \mathbb{R}^{+}_{\leq \alpha} \tag{3}$$

The $ReLU(\cdot,\cdot)$ function used in Equatoin (3) is a thresholded ReLU function, such that,

$$ReLU(x, \alpha) = \begin{cases} x & \text{if } 0 \leq x < \alpha \\ \alpha & \text{if } x \geq \alpha \end{cases} \tag{4}$$

In Equatoin (3), the $\mathcal{S}(\cdot,\cdot)$ represents a siamese network function, that receives two inputs. The framework contains a single shared DNN network $\mathcal{E}_\phi(\cdot)$ that map higher dimensional input to a lower dimension clusterable embeddings. The Euclidean distance of the embedding pairs is calculated and passed through the thresholded ReLU activation function derived in Equatoin (4). The threshold value is a cluster margin of α. Due to the threshold, the siamese architecture always generates outputs in the range $[0, \alpha]$. The L2 loss function is used to train the general AutoEmbedder architecture. The AutoEmbedder architecture is trained using an equal number of must-link and cannot link constraints for each data batch. However, in a triplet architecture, the problem is automatically solved, as each triplet contains a fusion of cannot link (negative) and can-link (positive) data.

3.7. Augmenting Training Data

Both types of cluster relationships (can-link and cannot-link) may contain faulty assumptions and pseudo labels considering the ground truth. Hence, a basic augmentation scheme is used to trick the DL network from overfitting erroneous cluster relationships. Although various augmentation techniques are available, we adhere to mixing noise with speech data for augmentation. For noise augmentation, we implement a basic formula that is,

$$Aug(x_i, noise, thres) = \{x_i^{'} | x_i^{'} = x_i \times (1 - thres) + noise \times thres\} \quad [0 \leq thres \leq 1] \tag{5}$$

Here, $Aug(\cdot,\cdot,\cdot)$ is a function that produces augmented speech data, which is inputted as x_i. The *thres* is a threshold used to define the ratio of mixing *noise* with speech data x_i. Augmenting noise with speech frames results in less-confusing the AutoEmbedder network in case of erroneous data pairs. Fusing noise may facilitate the architecture by ignoring faulty data pairs due to different noise situations. Moreover, augmenting data also results in data variation, and the network extracts more beneficial features from speech data. Algorithm 1 presents pseudocode of the pairwise training process.

Algorithm 1: AutoEmbedder training for speaker recognition.

Input: Dataset D containing speech frames, DL model with initial weights $\mathcal{E}_\phi$, Distance hyperparameter α, Training epochs E_p

Initialize siamese network, $\mathcal{S}_\phi(\cdot,\cdot) \leftarrow ReLU(||\mathcal{E}_\phi(\cdot) - \mathcal{E}_\phi(\cdot)||, \alpha)$

for $epoch \leftarrow 1$ **to** E_p **do**
  foreach $\mathcal{D}_{batch} \in \mathcal{D}$ **do**
    $\mathcal{X}$, $\mathcal{X}^{'}$, $\mathcal{Y} \leftarrow \{\}$, $\{\}$, $\{\}$
    $counter \leftarrow 0$
    foreach $x \in \mathcal{D}_{batch}$ **do**
      $\mathcal{X} \leftarrow$ append x in $\mathcal{X}$
      if $counter < |batch|/2$ **then**
        $\mathcal{X}^{'} \leftarrow$ randomly pick and append a can-link speech frame from $\mathcal{D}$
        $\mathcal{Y} \leftarrow$ append 0 in $\mathcal{Y}$
      else
        $\mathcal{X}^{'} \leftarrow$ randomly pick and append a cannot-link speech frame from $\mathcal{D}$
        $\mathcal{Y} \leftarrow$ append α in $\mathcal{Y}$
      $counter \leftarrow counter + 1$
    $\mathcal{X} \leftarrow$ randomly select half of the speech frames and augment them
    $\mathcal{X}^{'} \leftarrow$ randomly select half of the speech frames and augment them
    $S_\phi \leftarrow$ Train S_ϕ with $\mathcal{X}, \mathcal{X}^{'}, \mathcal{Y}$

4. Experiments and Evaluations

In this section, the proposed scheme is experimented based on the impurity of speech segmentation. As the architecture's target is to produce clusterable embedding, we use k-means to measure the purity of the clusters generated by the embedding system. Further, three popular metrics, Accuracy (ACC), normalized mutual information (NMI), and adjusted rand index (ARI), are used to measure clustering effectiveness. The metrics are calculated as demonstrated in [21], and are widely implemented to refer to the purity of clustering [33,39].

4.1. Experimental Setup

The datasets were segmented using a threshold of 16 decibels, implemented as a VAD. The audio streams have been processed with a sample rate of 16,000 Hz. For audio to spectrogram conversion, the parameters are set as described, size of fast-fourier transform: 191, window-size: 128, stride: 34, mel-scales: 100. Speech spectrograms are used as inputs to train the DL architectures.

4.2. Datasets

For experimentation, three speech datasets have been used. TIMIT [36] and LibriSpeech [37] are popular speech datasets for English language. Moreover, we use Bengali Automated Speech Recognition Dataset [38] to show the diversity of our approach for additional languages. Among the three datasets, TIMIT and LirbriSpeech datasets contain studio-grade audio speech. Bengali ASR dataset is crowdsourced hence, contains a diverse sound and noise variation. Table 2 illustrates some basic statistics of each dataset. Throughout the experiment, we abbreviate LibriSpeech and Bengali ASR dataset as LIBRI and ASR, respectively. As the training procedure augments noise with the speech frames, we use scalable noisy speech dataset [40]. The dataset contains diverse environmental noises, which helps the architectures to explore and relate speech features.

4.3. Result Analysis

Two methods are implemented, AutoEmbedder (pairwise architecture) and a triplet architecture, to analyze the speech embeddings based on the proposed strategy. However, apart from these two strategies, the currently famous speech vector methods do not hold to the training properties considered in the paper. They mostly follow a supervised learning or domain adaption strategy. Hence, they are disregarded in this experiment.

DenseNet121 [41], is used as a baseline architecture for both of the DL frameworks. Further, both models are connected with a dense layer consisting of 12 nodes. Therefore, both pairwise and triplet networks produce 12 dimension embedding vectors. L2-normalization is added on the output layer of the triplet network, as it is suggested that it increases the accuracy of the framework [42]. For AutoEmbedder architecture, the default l2-loss is implemented, whereas the triplet architecture is trained using semi-hard triplet loss. The training pipeline is illustrated in Figure 3.

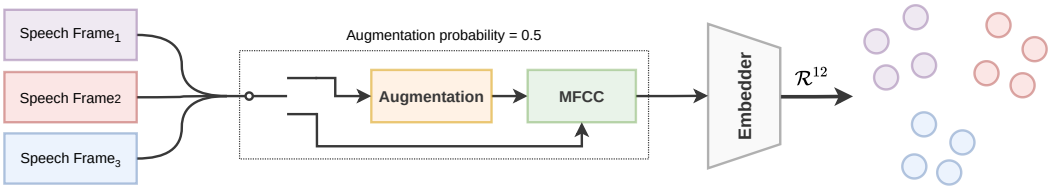


Figure 3. The training of both pairwise and triplet networks is done through the same data processing pipeline. Half of the inputs are randomly augmented and processed using MFCC. The MFCC of each frame is further passed to the DL frameworks.

The evaluation process guarantees that both architectures are trained using the same dataset/data-subset. As the training process is unsupervised, the architectures receive the same data for the training and testing process. However, for the training process, the labels are unknown and generated based on the paper's assumptions. We refer to such a dataset as a training dataset. By ground dataset, we refer to the same dataset that further considers the ground truth values. For training both frameworks, we used a batch size of 128. The training is conducted using Adam [43] optimizer with a learning rate of 0.0005. The learning rate and batch size were determined using a grid search. Although batch size 64 and 128 result in better evaluation, batch size 128 was considered due to faster computation.

The training phase's data processing includes heavy computational complexity, including online noise augmentation and spectrogram conversion. Each dataset is randomly augmented with a threshold range of $[0, 0.07]$ for the augmentation process. Further, computing ACC, NMI, and ARI metrics require quadratic time complexity. Hence, we limit the number of speakers to 150. Instead of training on the overall dataset, we train on a sub-set of data, where each speaker contains a speech of 10 s. For testing the ground truth data, a random selection of 2-s speech is selected for each speaker. To inquire the architectures properly, we scramble the training dataset's pseudo labels, that produces a can-link impurity in the dataset labels. Hence, we use the impurity ratio as a training data situation to illustrate the rate of impure cluster assignments on the training data pseudo-labels. The models were kept training until the performance on ground truth labels did not improve within the previous 1500 epochs.

Figure 4 illustrates a benchmark of the triplet and pairwise architectures while training on three different datasets, with speakers = 25 and impurity = 0. The triplet architectures smoothly learn from the training data and greatly overfits on the augmented training data. The ground dataset benchmark is also as expected since the triplet architectures' correctness first increases and gradually decreases due to overfitting. Hence, from the visualization, it can be acknowledged that the triplet network only memorizes the speech features concerning the pseudo labels assigned to them.

On the contrary, the pairwise architecture generates a satisfactory performance, with some irregularities. In general, deep learning architectures produce higher accuracy on

the training dataset than validation dataset. However, in such a case, the ground dataset's performance is mostly more elevated than the training dataset. Yet, the performance on the ground datasets generally decreases after 400 epochs. As the number of speakers is small, the architecture easily gets overfitted on the training dataset. Further increasing the number of speakers to 50 reduces the overfitting on training data, as illustrated in Figure 5. The triplet architecture still overfits on the training data's pseudo label, whereas, the pairwise architecture gives a balanced performance on the ground dataset.

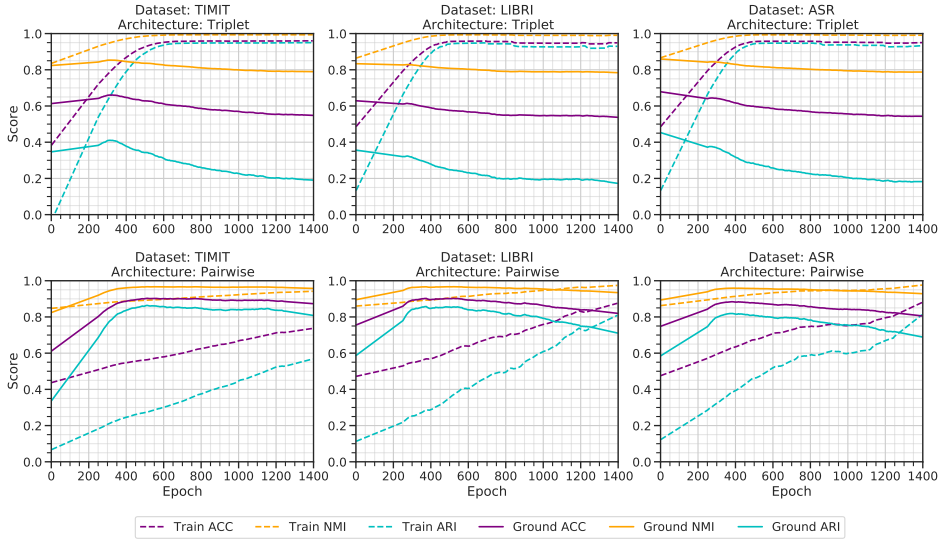


Figure 4. The graphs illustrated in the first row visualizes the metrics on training and ground dataset containing 25 speakers with impurity = 0 for triplet architecture. The lower row envisions the same for the pairwise architecture. Each column represents benchmarks carried on a single dataset.

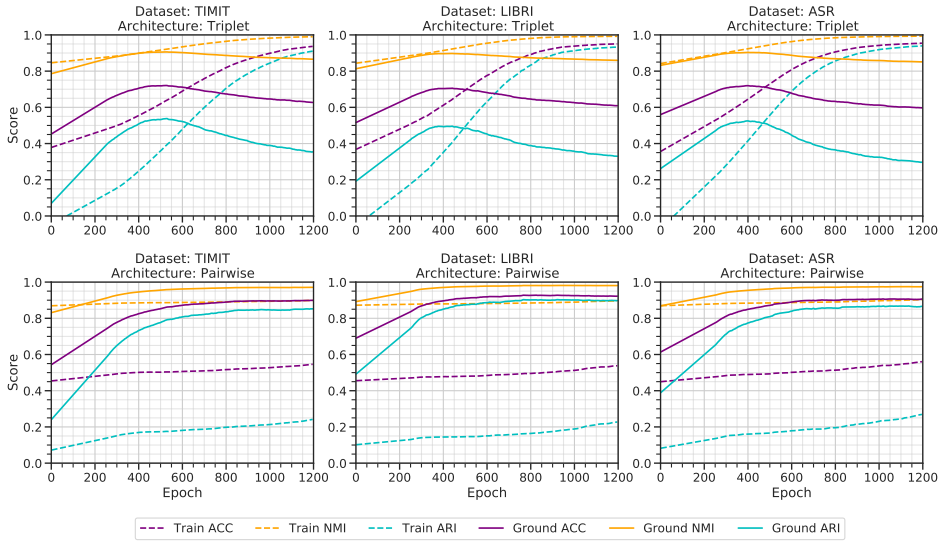


Figure 5. The graph illustrates metrics on training and ground dataset containing 50 speakers with impurity = 0.

Increasing the impurity of the inter-connection of the training data reduces the performance of the architectures. Figures 6 and 7 illustrates benchmarks conducted with impurity = 0.05 and impurity = 0.1 while considering speakers = 50. The triplet architecture still overfits on the training architecture. In contrast, pairwise architecture slowly memorizes the training dataset. Yet, it holds a marginal exactness on the ground data before overfitting on the training data.

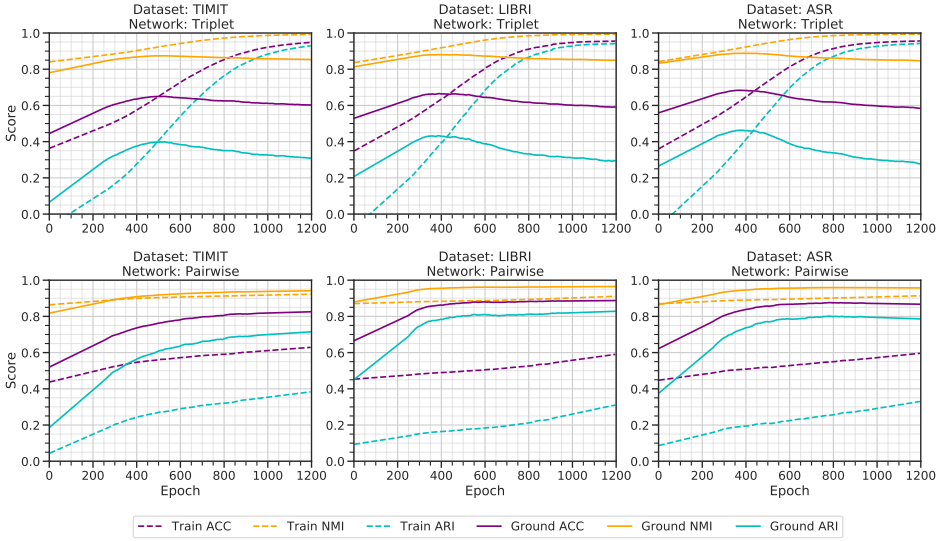


Figure 6. The graph illustrates metrics on training and ground dataset containing 50 speakers with impurity = 0.05.

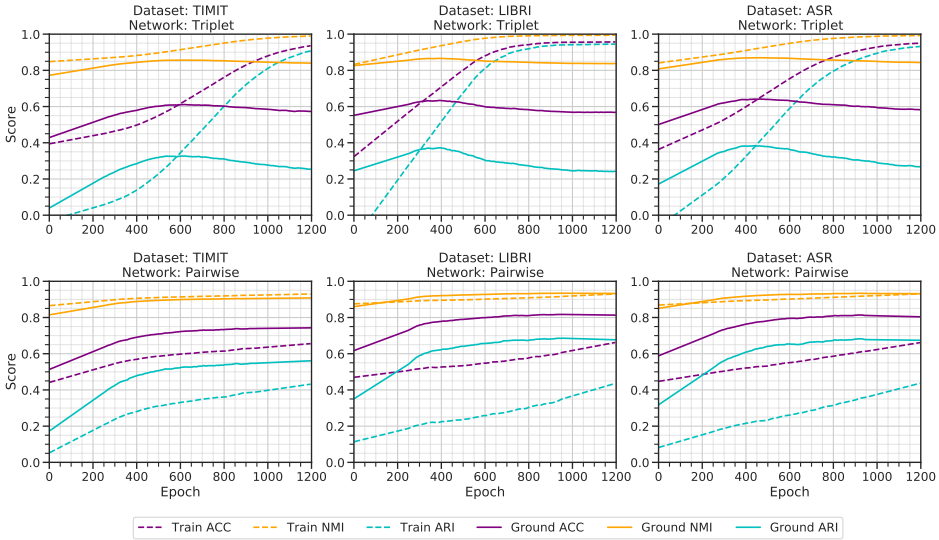


Figure 7. The graph illustrates metrics on training and ground dataset containing 50 speakers with impurity = 0.1.

Triplet architecture, trained over semi-hard triplet loss strictly targets the positive points (based on the anchor) and minimizes the distance between the anchor and positive data. For negative points, the loss function also strictly distance the embeddings. As the loss function heavily maintains the criteria mentioned above, the architecture gets overfitted on the hypothetical constrains disregarding the actual feature-dependent relations.

In contrast, AutoEmbedder architecture learns to extract features rather than being overfitted in the training data. The reasoning lies in the training strategy of the network. The l2-loss does not strictly consider learning the hypothetical constraints and learns for each batch of data aggregately. As the architecture is not strictly supervised using the loss function, it obtains the feature similarity of audio data. Therefore the architecture re-clusters the data on hyperspace based on feature similarity.

We further investigate with AutoEmbedder based pairwise architecture with various speaker and impurity condition. Tables 3–5 illustrates the metrics on training and ground dataset for TIMIT, LIBRI and, ASR datasets, respectively. The table illustrates a detailed overview of the performance variation based on the number of speakers and impurity in the training dataset. The pairwise architecture maintains a marginal performance with

impurity = 0 on every dataset. However, increasing the number of speakers results in reducing the performance of the architecture. On the contrary, increasing the impurity of the speech segment further reduces the performance of the architecture. A small fluctuation is observed for LIBRI and ASR dataset while the number of speakers is kept on 25 and 50. Increasing the number of speakers from 25 to 50 causes an increase in accuracy, which is inconsistent.

Table 3. The table benchmarks the pairwise architecture in TIMIT dataset with four groups of speakers, 25, 50, 100, and 150. For each group of speakers, the table also considers three segmentation impurities, 0, 0.05, and 0.1 to illustrate the shortcomings of incorrect segmentation, for fully unsupervised speaker recognition strategy.

	Impurity = 0			**Impurity = 0.05**			**Impurity = 0.1**		
	ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
Speakers = 25									
Train	0.772	0.950	0.627	0.868	0.974	0.795	0.893	0.980	0.842
Ground	0.934	0.979	0.919	0.886	0.958	0.822	0.777	0.924	0.630
Speakers = 50									
Train	0.663	0.930	0.436	0.659	0.930	0.435	0.678	0.935	0.464
Ground	0.924	0.979	0.898	0.849	0.948	0.755	0.764	0.915	0.623
Speakers = 100									
Train	0.509	0.899	0.175	0.523	0.904	0.208	0.590	0.920	0.305
Ground	0.889	0.973	0.847	0.809	0.948	0.723	0.719	0.914	0.544
Speakers = 150									
Train	0.471	0.861	0.117	0.500	0.902	0.155	0.519	0.907	0.194
Ground	0.815	0.955	0.734	0.718	0.924	0.645	0.620	0.896	0.480

Table 4. The table benchmarks the pairwise architecture in LIBRI dataset with four groups of speakers, 25, 50, 100, and 150. For each group of speakers, the table also considers three segmentation impurities, 0, 0.05, and 0.1 to illustrate the shortcomings of incorrect segmentation, for fully unsupervised speaker recognition strategy.

	Impurity = 0			**Impurity = 0.05**			**Impurity = 0.1**		
	ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
Speakers = 25									
Train	0.936	0.989	0.911	0.954	0.992	0.940	0.956	0.993	0.943
Ground	0.946	0.983	0.935	0.866	0.948	0.808	0.806	0.918	0.673
Speakers = 50									
Train	0.730	0.947	0.555	0.741	0.950	0.576	0.755	0.953	0.594
Ground	0.951	0.989	0.933	0.906	0.971	0.857	0.863	0.957	0.778
Speakers = 100									
Train	0.482	0.891	0.133	0.511	0.900	0.186	0.588	0.920	0.309
Ground	0.924	0.984	0.921	0.885	0.970	0.831	0.840	0.949	0.700
Speakers = 146									
Train	0.490	0.780	0.109	0.493	0.900	0.150	0.515	0.906	0.190
Ground	0.932	0.987	0.916	0.797	0.949	0.678	0.713	0.923	0.607

The architecture requires a sufficient number of speech variations from users to explore the proper feature relationship between speech frames. Limiting the number of speakers to 25 caused the interpretation of speech to be reduced. Hence, the architecture struggles to find better speech relations, and lessened performance is observed. Increasing the number

of speakers to 50 balances the speech variations in the training data and causes an increase in accuracy.

Table 5. The table benchmarks the pairwise architecture in ASR dataset with four groups of speakers, 25, 50, 100, and 150. For each group of speakers, the table also considers three segmentation impurities, 0, 0.05, and 0.1 to illustrate the shortcomings of incorrect segmentation, for fully unsupervised speaker recognition strategy.

	Impurity = 0			**Impurity = 0.05**			**Impurity = 0.1**		
	ACC	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**	**ACC**	**NMI**	**ARI**
Speakers = 25									
Train	0.920	0.985	0.882	0.945	0.990	0.923	0.954	0.992	0.941
Ground	0.909	0.969	0.869	0.854	0.941	0.768	0.789	0.907	0.643
Speakers = 50									
Train	0.771	0.955	0.617	0.614	0.918	0.355	0.666	0.933	0.448
Ground	0.930	0.983	0.912	0.903	0.966	0.835	0.841	0.948	0.751
Speakers=100									
Train	0.506	0.899	0.179	0.536	0.906	0.224	0.561	0.914	0.270
Ground	0.906	0.977	0.871	0.836	0.956	0.753	0.727	0.918	0.647
Speakers = 150									
Train	0.487	0.880	0.149	0.486	0.900	0.150	0.536	0.912	0.226
Ground	0.885	0.973	0.836	0.714	0.926	0.687	0.671	0.908	0.568

5. Discussion

The pairwise architecture with training strategy results in a good performance in the speaker recognition process. However, throughout the investigation, the architecture tends to have some issues that have to be considered. Firstly, training the architecture with lesser speech variation causes overfitting, observed while keeping $speaker = 25$. Secondly, as the augmentation procedure fuses noises, speech data with excessive noises may not generate a good result. The degradation of performance due to excessive noise is a common concern, and by using a denoising mechanism, the situation can be handled [2]. Further, as the system is fully segmentation dependent, the target lies in developing an optimal audio segmentation procedure. Resolving these challenges would benefit the architecture for a wide range of speaker recognition and evaluation usage.

Apart from the limitations, the u-vector strategy requires no pre-training on large speaker datasets, which is often observed in i-vector, d-vector, and x-vector [7]. Further, the u-vector strategy requires comparatively less per-speaker data than the other embedding strategies mentioned above. In the case of augmentations, u-vector architecture does not require any labeled data, which is primarily observed in generative speaker recognition architectures [44]. In an overall perspective, the u-vector mitigates the requirement of labeled data to a minimum.

Aside from implementing u-vectors in an unsupervised and semi-supervised manner, the strategy can be implemented in self-supervised learning. Although self-supervised learning has its various forms based on the domain, u-vector aligns with contrastive self-supervised learning strategies [45]. In general, self-supervised learning learns better model representation (weights of a model) from unlabeled data. Further, the trained model is used to train on a small quantity of labeled data. U-vectors can be used to initialize model representations in the first stage of self-supervised learning. Further, a classification method/layer can be equipped with the model for training the model on labeled data.

Finally, industrial systems often can not label every data. However, if a large labeled dataset is required, then building a model can become costly. Therefore, unsupervised, semi/self-supervised learning in speaker recognition systems is expected to produce low-cost and attainable systems.

6. Conclusions

The paper introduces a system of generating clusterable speech embedding based on the speakers, namely u-vector. The policy of the architecture deals with pseudo labels and trained from unlabeled datasets. The procedure is suitable for both semi-supervised and unsupervised training strategies. We evaluate such strategies with two appropriate deep learning architectures: pairwise and triplet. In the perspective of unlabeled data, the architecture performs at an acceptable rate concerning the number of speakers and speech segmentation errors. However, the method requires clean speech, and robust segmentation techniques to properly construct clusterable u-vectors, depending on speaker variations. We strongly believe that such an in-depth and hypothetical strategy of generating pseudo labels to train speaker recognition models would help researches develop new schemes.

Author Contributions: Conceptualization, M.F.M., A.Q.O., M.M.M., M.A.H., M.R.I. and Y.W.; Data curation, M.F.M. and A.Q.O.; Formal analysis, M.F.M., A.Q.O., M.M.M., M.A.H.; Investigation, M.F.M., M.R.I. and Y.W.; Methodology, M.F.M., A.Q.O., M.M.M., M.A.H., M.R.I. and Y.W.; Software, M.F.M. and A.Q.O.; Supervision, M.M.M., M.A.H., M.R.I. and Y.W.; Validation, M.M.M., M.A.H., M.R.I. and Y.W.; Visualization, M.F.M., M.M.M., M.A.H., M.R.I. and Y.W.; Writing–original draft, M.F.M., A.Q.O.; Writing–review & editing, M.M.M., M.A.H., M.R.I. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to thank Bangladesh University of Business & Technology (BUBT), University of Asia Pacific (UAP), and University of Aizu (UoA) for supporting this research. Also, special thanks to the Advanced Machine Learning lab, BUBT; Computer Vision & Pattern Recognition Lab, UAP; Database System Lab, UoA; for giving facilities to research and publish.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Graves, A.; Fernández, S.; Gomez, F.; Schmidhuber, J. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25–29 June 2006; pp. 369–376.
2. Azarang, A.; Kehtarnavaz, N. A review of multi-objective deep learning speech denoising methods. *Speech Commun.* **2020**, *122*, 1–10. [CrossRef]
3. Ning, Y.; He, S.; Wu, Z.; Xing, C.; Zhang, L.J. A review of deep learning based speech synthesis. *Appl. Sci.* **2019**, *9*, 4050. [CrossRef]
4. Fayek, H.M.; Lech, M.; Cavedon, L. Evaluating deep learning architectures for Speech Emotion Recognition. *Neural Netw.* **2017**, *92*, 60–68. [CrossRef] [PubMed]
5. Kamper, H.; Livescu, K.; Goldwater, S. An embedded segmental k-means model for unsupervised segmentation and clustering of speech. In Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Okinawa, Japan, 16–20 December 2017; pp. 719–726.
6. O'Brien, M.G.; Derwing, T.M.; Cucchiarini, C.; Hardison, D.M.; Mixdorff, H.; Thomson, R.I.; Strik, H.; Levis, J.M.; Munro, M.J.; Foote, J.A.; others. Directions for the future of technology in pronunciation research and teaching. *J. Second. Lang. Pronunciation* **2018**, *4*, 182–207. [CrossRef]
7. Ohi, A.Q.; Mridha, M.; Hamid, M.A.; Monowar, M.M. Deep Speaker Recognition: Process, Progress, and Challenges. *IEEE Access* **2021**, *9*, 89619–89643. [CrossRef]
8. Kabir, M.M.; Mridha, M.; Shin, J.; Jahan, I.; Ohi, A.Q. A Survey of Speaker Recognition: Fundamental Theories, Recognition Methods and Opportunities. *IEEE Access* **2021**, *9*, 79236–79263. [CrossRef]
9. Tiwari, V. MFCC and its applications in speaker recognition. *Int. J. Emerg. Technol.* **2010**, *1*, 19–22.
10. Chowdhury, A.; Ross, A. Fusing MFCC and LPC features using 1D triplet CNN for speaker recognition in severely degraded audio signals. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 1616–1629. [CrossRef]
11. Ravanelli, M.; Bengio, Y. Speaker recognition from raw waveform with sincnet. In Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT), Athens, Greece, 18–21 December 2018; pp. 1021–1028.
12. Chagas Nunes, J.A.; Macêdo, D.; Zanchettin, C. AM-MobileNet1D: A Portable Model for Speaker Recognition. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8. [CrossRef]

13. Garcia-Romero, D.; Espy-Wilson, C.Y. Analysis of i-vector length normalization in speaker recognition systems. In Proceedings of the Twelfth Annual Conference of the International Speech Communication Association, Florence, Italy, 27–31 August 2011.
14. Variani, E.; Lei, X.; McDermott, E.; Moreno, I.L.; Gonzalez-Dominguez, J. Deep neural networks for small footprint text-dependent speaker verification. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 4052–4056.
15. Snyder, D.; Garcia-Romero, D.; Sell, G.; Povey, D.; Khudanpur, S. X-vectors: Robust dnn embeddings for speaker recognition. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 5329–5333.
16. Bai, Z.; Zhang, X.L. Speaker recognition based on deep learning: An overview. *Neural Netw.* **2021**, *140*, 65–99. [CrossRef]
17. Wang, Q.; Rao, W.; Sun, S.; Xie, L.; Chng, E.S.; Li, H. Unsupervised domain adaptation via domain adversarial training for speaker recognition. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 4889–4893.
18. Garcia-Romero, D.; McCree, A.; Shum, S.; Brummer, N.; Vaquero, C. Unsupervised domain adaptation for i-vector speaker recognition. In Proceedings of the Odyssey: The Speaker and Language Recognition Workshop, Tokyo, Japan, 17–21 May 2014; Volume 8.
19. Garcia-Romero, D.; Zhang, X.; McCree, A.; Povey, D. Improving speaker recognition performance in the domain adaptation challenge using deep neural networks. In Proceedings of the 2014 IEEE Spoken Language Technology Workshop (SLT), South Lake Tahoe, NV, USA, 7–10 December 2014; pp. 378–383.
20. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
21. Ohi, A.Q.; Mridha, M.; Safir, F.B.; Hamid, M.A.; Monowar, M.M. AutoEmbedder: A semi-supervised DNN embedding system for clustering. *Knowl.-Based Syst.* **2020**, *204*, 106190. [CrossRef]
22. Zhang, C.; Koishida, K.; Hansen, J.H. Text-independent speaker verification based on triplet convolutional neural network embeddings. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2018**, *26*, 1633–1644. [CrossRef]
23. Campbell, W.M.; Sturim, D.E.; Reynolds, D.A. Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Process. Lett.* **2006**, *13*, 308–311. [CrossRef]
24. Kenny, P.; Boulianne, G.; Ouellet, P.; Dumouchel, P. Joint factor analysis versus eigenchannels in speaker recognition. *IEEE Trans. Audio Speech Lang. Process.* **2007**, *15*, 1435–1447. [CrossRef]
25. Dehak, N.; Kenny, P.J.; Dehak, R.; Dumouchel, P.; Ouellet, P. Front-end factor analysis for speaker verification. *IEEE Trans. Audio Speech Lang. Process.* **2010**, *19*, 788–798. [CrossRef]
26. Molla, K.; Hirose, K. On the effectiveness of MFCCs and their statistical distribution properties in speaker identification. In Proceedings of the 2004 IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems, 2004.(VCIMS), Boston, MA, USA, 12–14 July 2004; pp. 136–141.
27. Ioffe, S. Probabilistic linear discriminant analysis. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 531–542.
28. Chen, Y.h.; Lopez-Moreno, I.; Sainath, T.N.; Visontai, M.; Alvarez, R.; Parada, C. Locally-connected and convolutional neural networks for small footprint speaker recognition. In Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association, Dresden, Germany, 6–10 September 2015.
29. Snyder, D.; Garcia-Romero, D.; Povey, D.; Khudanpur, S. Deep Neural Network Embeddings for Text-Independent Speaker Verification. In Proceedings of the Interspeech, Stockholm, Sweden, 20–24 August 2017; pp. 999–1003.
30. Redko, I.; Morvant, E.; Habrard, A.; Sebban, M.; Bennani, Y. *Advances in Domain Adaptation Theory*; Elsevier: Amsterdam, The Netherlands, 2019.
31. Spyrou, E.; Mathe, E.; Pikramenos, G.; Kechagias, K.; Mylonas, P. Data Augmentation vs. Domain Adaptation—A Case Study in Human Activity Recognition. *Technologies* **2020**, *8*, 55. [CrossRef]
32. Guo, Q.; Feng, W.; Zhou, C.; Huang, R.; Wan, L.; Wang, S. Learning dynamic siamese network for visual object tracking. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1763–1771.
33. Ren, Y.; Hu, K.; Dai, X.; Pan, L.; Hoi, S.C.; Xu, Z. Semi-supervised deep embedded clustering. *Neurocomputing* **2019**, *325*, 121–130. [CrossRef]
34. Tan, Z.H.; Dehak, N.; others. rVAD: An unsupervised segment-based robust voice activity detection method. *Comput. Speech Lang.* **2020**, *59*, 1–21. [CrossRef]
35. Brent, M.R. Speech segmentation and word discovery: A computational perspective. *Trends Cogn. Sci.* **1999**, *3*, 294–301. [CrossRef]
36. Garofolo, J.S.; Lamel, L.F.; Fisher, W.M.; Fiscus, J.G.; Pallett, D.S. DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. *STIN* **1993**, *93*, 27403.
37. Panayotov, V.; Chen, G.; Povey, D.; Khudanpur, S. Librispeech: An asr corpus based on public domain audio books. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, 19–24 April 2015; pp. 5206–5210.
38. Kjartansson, O.; Sarin, S.; Pipatsrisawat, K.; Jansche, M.; Ha, L. Crowd-Sourced Speech Corpora for Javanese, Sundanese, Sinhala, Nepali, and Bangladeshi Bengali. In Proceedings of the 6th International Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU), St Petersburg, Russia, 14–16 May 2018; pp. 52–55.

39. Yang, X.; Deng, C.; Zheng, F.; Yan, J.; Liu, W. Deep spectral clustering using dual autoencoder network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4066–4075.
40. Reddy, C.K.; Beyrami, E.; Pool, J.; Cutler, R.; Srinivasan, S.; Gehrke, J. A Scalable Noisy Speech Dataset and Online Subjective Test Framework. *Proc. Interspeech 2019* **2019**, 1816–1820.
41. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
42. Hermans, A.; Beyer, L.; Leibe, B. In defense of the triplet loss for person re-identification. *arXiv* **2017**, arXiv:1703.07737.
43. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
44. Pal, M.; Kumar, M.; Peri, R.; Park, T.J.; Kim, S.H.; Lord, C.; Bishop, S.; Narayanan, S. Meta-learning with latent space clustering in generative adversarial network for speaker diarization. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 1204–1219. [CrossRef] [PubMed]
45. Jaiswal, A.; Babu, A.R.; Zadeh, M.Z.; Banerjee, D.; Makedon, F. A survey on contrastive self-supervised learning. *Technologies* **2021**, *9*, 2. [CrossRef]