acoustics

MDPI

Article

On Training Targets and Activation Functions for Deep Representation Learning in Text-Dependent Speaker Verification

Achintya Kumar Sarkar [1,*] and Zheng-Hua Tan [2,*]

[1] Indian Institute of Information Technology, Sri City/Chittoor 517646, India
[2] Department of Electronic Systems, Aalborg University, 9220 Aalborg, Denmark
* Correspondence: sarkar.achintya@gmail.com (A.K.S.); zt@es.aau.dk (Z.-H.T.)

Abstract: Deep representation learning has gained significant momentum in advancing text-dependent speaker verification (TD-SV) systems. When designing deep neural networks (DNN) for extracting bottleneck (BN) features, the key considerations include training targets, activation functions, and loss functions. In this paper, we systematically study the impact of these choices on the performance of TD-SV. For training targets, we consider speaker identity, time-contrastive learning (TCL), and auto-regressive prediction coding, with the first being supervised and the last two being self-supervised. Furthermore, we study a range of loss functions when speaker identity is used as the training target. With regard to activation functions, we study the widely used sigmoid function, rectified linear unit (ReLU), and Gaussian error linear unit (GELU). We experimentally show that GELU is able to reduce the error rates of TD-SV significantly compared to sigmoid, irrespective of the training target. Among the three training targets, TCL performs the best. Among the various loss functions, cross-entropy, joint-softmax, and focal loss functions outperform the others. Finally, the score-level fusion of different systems is also able to reduce the error rates. To evaluate the representation learning methods, experiments are conducted on the RedDots 2016 challenge database consisting of short utterances for TD-SV systems based on classic Gaussian mixture model-universal background model (GMM-UBM) and i-vector methods.

Keywords: training targets; activation functions; loss functions; bottleneck features; text-dependent speaker verification

Citation: Sarkar, A.K.; Tan, Z.-H. On Training Targets and Activation Functions for Deep Representation Learning in Text-Dependent Speaker Verification. *Acoustics* **2023**, *3*, 693–713. https://doi.org/10.3390/acoustics5030042

Academic Editor: Jian Kang

Received: 9 May 2023
Revised: 7 July 2023
Accepted: 11 July 2023
Published: 17 July 2023

1. Introduction

Speaker verification (SV) is an authentication technique to verify a person using their speech sample. It is a binary classification system. Due to its non-invasive nature, SV has attracted great interest in many authentication services such as voice mail, home automation, computer login, online resource access, IoT, etc. Depending on the constraint of lexicon or phonetic content in the speech sample, SV systems can be broadly categorized as text-independent (TI) or text-dependent (TD). In TD-SV, speakers utter the same pass-phrase during both the enrollment and test phases to maintain the matched phonetic content. On the other hand, the speakers are free to speak any text during the training and test phases in TI-SV, i.e., there is no constraint to speak the same pass-phrase during both training and testing. Therefore, TD-SV is able to yield much lower error rates than TI-SV, especially when using short utterances. Additionally, the response time of TD-SV, due to the need for short utterances only, is much shorter compared to TI-SV, which makes it attractive for real-time applications.

A variety of methods were proposed in the literature to improve the performance of TD-SV. These methods are grouped into feature domain [1], model domain [2,3], and score domain [4]. In the feature domain, one type of feature includes engineered short-time cepstral features, such as Mel-frequency cepstral coefficients (MFCC) [5], power normalized

cepstral coefficients [6], and perceptual linear prediction [7]. Another contains learned bottleneck (BN) features, which are derived from deep neural networks (DNN) where a DNN is trained to discriminate or predict a chosen target. Afterward, the frame-level output of a particular hidden layer is projected onto a low dimensional space to obtain BN features [8]. The low dimensional space is usually found using principal component analysis (PCA). In [9], audio segments of variable lengths are represented by fixed-length vectors using the concept of the sequence-to-sequence autoencoder. A fusion embedding network is proposed in [10] to combine the advantage of TI-SV and TD-SV in joint learning. A multi-task learning network, which is based on a phoneme-aware and channel-wise attentive learning strategy, is proposed for TD-SV to disentangle the speaker and text information [11]. A memory layer and multi-head attention mechanism-based DNN is proposed to improve the efficiency of TD-SV systems in [12]. A synthesis-based data augmentation method is introduced in [13] to increase the speakers' and text-controlled speech data for TD-SV. In this work, we focus on the feature domain, in particular, deep features at frame level.

In training DNNs for feature extraction, various training targets are used, and examples are speakers [8], phones [1], pass-phrases [8], senones [14], time-contrastive learning targets [1], and auto-regressive prediction coding (APC) targets [15]. Most of the BN feature extraction methods require label information such as speaker identities, pass-phrases, and phones. The generation of label information can be time-consuming and expensive. As an alternative, self-supervised and semi-supervised learning is very appealing, which can leverage a large amount of unlabeled data available in the real world. Recently, APC [16] and time-contrastive learning (TCL) [1] BN features were introduced for speech representation learning for SV. In APC-BN, a DNN is trained with the objective to predict the future feature vector using the current and past frames. Then, the last hidden layer is used for BN feature extraction. Given that the objective of APC is to predict the content of the next frame, it is unknown whether the last hidden layer is the optimal choice. On the other hand, TCL uniformly divides the speech signal into a number of predefined segments, and then, the frames within a particular segment are assigned one same class label. Afterward, a DNN is trained to discriminate these classes for BN feature extraction. TCL aims to capture the temporal/phonetic information in the speech signal in a self-supervised manner and is shown to be very useful for TD-SV. As both the recently proposed APC and TCL BN features are extracted in a self-supervised manner, it is of interest and relevance to compare their performance and behavior in the same framework.

In addition to the selection of training targets, the other essential choices in DNN design include activation functions and loss functions, which are both key elements for DNN training. A loss function measures the error between the network output and the desired target, and in error back-propagation training, the derivative of the loss function is used to guide the training through the gradient descent approach. Various loss functions were introduced in the literature for improved representation learning for such tasks as speech recognition, speaker verification, and image classification, and the examples are joint softmax-center loss [17], modified-softmax [18], arcFace [19], focal [20], orthogonal softmax layer (OSL) [21], triplet-loss [22], the simple framework for contrastive learning (SimCLR) [23], and cross-entropy.

Activation functions, on the other hand, control the output of DNN hidden neurons, as well as the gradient contribution during the error back-propagation process for network parameter optimization. Among others, sigmoid [24,25] and ReLU [26] are most widely used in the state-of-the-art systems such as speaker recognition [27–30] and language recognition [31,32], speech recognition [33,34], prosodic representation [35], and image processing [19,22,23].

Although widely used, the sigmoid function suffers from a major problem, namely, gradient vanishing. This is because the function squishes the input space into a range between 0 and 1, and hence, a large change in input may have a small change in the output, leading to a very small derivative. The multiplication through hidden layers in

back-propagation decreases the gradient exponentially. In the end, initial layers are not updated properly, and thus, the model is not trained effectively and lacks in generalization ability [36,37]. To avoid the vanishing gradient problem, the ReLU activation function is widely used as well. As it preserves the large dynamic range of input in the output (from 0 to maximum), as compared to the sigmoid function, it provides a better generalization performance and is simple. As per [38], the sigmoid function is ineffective for training DNNs due to the gradient vanishing problem, and ReLU lacks in probabilistic interpretation and, thus, requires stochastic regularization for better training of DNN. To combine stochastic regularization with a deterministic activation function, the GELU activation function is introduced in [38]. It is shown in [38] that GELU outperforms ReLU, the exponential linear unit (ELU) in different tasks including speech recognition, language processing, and computer vision. For extracting speaker embeddings, GELU is found being used in Transformers and multi-layer perceptron-based speaker verification networks (MLP-SVNet) systems [39,40].

The methods for BN feature extraction in TD-SV usually consider sigmoid activation function, and if discriminative loss function is needed, cross-entropy is used for discriminating, e.g., speakers, pass-phrases, senons, and TCL segments. The focus is on defining training targets, while loss functions and activation functions are significantly under-explored. Therefore, we aim at filling in this gap in this work, namely, to study the effect of different loss and activation functions, in connection with training targets, for BN feature extraction in TD-SV.

The contributions of this work are five-fold. First, we systematically study the impact of training targets, activation functions, and loss functions for the extraction of BN features on the performance of TD-SV in one joint framework, i.e., the evaluation of different training targets and activation and loss functions is based on the same DNN structure for BN feature extraction and the same TD-SV back-end and task. Second, we investigate ReLU and GELU activation functions for BN feature extraction for TD-SV and compare them with the commonly used sigmoid function in this context. Third, we study the impact of a set of loss functions on TD-SV performance. Fourth, we compare the performance of speaker-discriminant (Spkr) BN, TCL-BN, and APC-BN features, with the first being supervised and the last two being self-supervised. Finally, we analyze the performance of BN features extracted from different hidden layers and the performance of the score-level fusion of TD-SV systems based on different features.

We show that (1) both ReLU and GELU are able to reduce TD-SV error rates significantly compared with the commonly used sigmoid function in most cases, and GELU generally performs the best; (2) cross-entropy, joint-softmax, and focal loss functions outperform the others; (3) TCL is the best-performing training target; and (4) the fusion of different systems in the score domain further reduces the error rate. For the TD-SV system, we consider two well-known state-of-the-art techniques: the Gaussian mixture model-universal background model (GMM-UBM) [41] and i-vector [2] with scoring based on supervised probabilistic linear discriminate analysis (PLDA) training [42,43].

The paper is organized as follows. Section 2 presents three training targets and their corresponding BN features. Sections 3 and 4 introduce loss functions and activation functions, respectively. Section 5 presents the GMM-UBM and i-vector methods used for speaker modeling. The experimental setup is described in Section 6. Section 7 provides results and discussions. Finally, the paper is concluded in Section 8.

2. BN Features and Their Training Targets

We consider both supervised and self-supervised learning methods. The former method uses manually generated labels, while the latter derives the training target from the data itself without using human labels. More specifically, for supervised learning, speaker identities are used as the training targets, and for self-supervised learning, TCL and APC training targets are used. After training, the DNN, frame-level output from a particular hidden layer is projected onto a low dimensional space as per [8] to obtain BN features.

The low dimensional feature is suitable for the GMM- and i-vector-based classifiers and aligns with the dimension of the cepstral feature for a fair comparison. Figure 1 shows a block diagram of extracting BN features from the second hidden layer of a DNN.

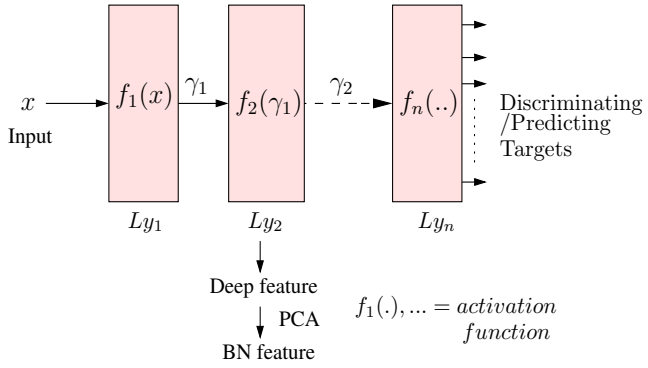

Figure 1. A DNN system, trained to discriminate or predict targets, for generating BN features using the second hidden layer.

The proposed approach differs from the D-vector approach [44], where each speech utterance/segmented utterance is represented by a *single vector*, which is calculated by averaging the frame level output from all hidden layers of the DNN. In our method, each frame level output from a particular (single) hidden layer is projected onto low dimensional space via PCA, i.e., each frame yields an output that is used as a feature for speaker recognition.

2.1. Spkr-BN

This is a supervised feature extraction method where a feed-forward DNN is trained using speaker identity labels as the training target to discriminate the speakers at the output layer [8,27,28]. The generated BN feature is called *Spkr-BN*. The other studies, e.g., in [1], also consider a multitask objective function to discriminate both speakers and pass-phrases at the output layer of DNN for BN feature extraction. However, the performance of the obtained BN is close to Spkr-BN. Utterance-level embeddings (into fixed-dimensional space) based on a convolutional neural network (CNN) can be found in [45], which differs from frame-level speech representations studied in the context of the present work.

2.2. TCL-BN [1]

This is a self-supervised learning method with certain similarities to contrasting learning methods, e.g., contrastive predictive coding [46] and HuBERT [47] methods. In the TCL-BN method, each speech signal is uniformly segmented into a fixed number of segments, and then, the data points within a particular segment are assigned one same class label as the training target; the first segment of a signal belongs to class one, the second segment to class two, and so on. These generated targets are then used for the training of a DNN with cross-entropy loss functions, and the derived feature is called uTCL-BN. The objective is to capture temporal information (e.g., discrimination of phonetic contents or words) in the speech signal in an unsupervised manner (without using automatic speech recognition or any manual label information).

For the c number of classes in uTCL, each speech signal is uniformly partitioned into c segments. The frames within the nth segments are assigned class label n as:

$$\underbrace{(x_1, ..., x_M)}_{class\ 1}, \ldots, \underbrace{(x_{iM+1}, ..., x_{iM+M})}_{...}, \ldots, \underbrace{(x_{(c-1)M+1}, ..., x_{cM})}_{class\ c} \quad (1)$$

where x denotes the frame-based feature vector.

In another case, speech signals are first randomly shuffled and then concatenated into a single long-duration stream. The stream is split into chunks of M frames, each with

$M = 6$. At a time, n chunks are taken and assigned the class labels $1, 2, \ldots, n$, respectively, which is repeated until all chunks are taken, and then, a DNN is trained to discriminate classes for extracting BN features called sTCL-BN. In this study, we consider the value of $n = 10$ as per [1].

2.3. APC-BN [15]

In this self-supervised learning method, a DNN encoder is trained to output a sequence $(o_1, o_2, \ldots, o_N)$ as a prediction of a given target sequence $(t_1, t_2, \ldots t_N)$ that is generated by right-shifting the input sequence $(x_1, x_2, \ldots, x_N)$ of t_n time steps. Then, the objective function is defined as the $\ell 1$ loss between them

$$\sum_{i=1}^{N-t_n} |t_i - o_i|, \quad t_i = x_{i+t_n}. \tag{2}$$

which is to be minimized.

The output from a particular hidden layer of the DNN for a given utterance at frame level is extracted to obtain the high dimensional deep APC feature for text-independent speaker verification and identification [15]. In [16], the deep APC feature vectors are further projected onto a low dimensional space to obtain the APC-BN feature for TD-SV.

3. Loss Functions

In this section, we describe a set of loss functions that were successfully applied to various application domains and are used in this work for training DNNs to extract bottleneck features. In particular, we focus on loss functions for classification. Note that, in the case of APC-BN, the $\ell 1$ loss is used for prediction/regression, as already presented in the section above.

3.1. Cross-Entropy

In this method, a feed-forward DNN is trained to discriminate the classes at the output layer with cross-entropy (CE) as the loss function

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^{N} y_i \log p(x_i, \theta) \tag{3}$$

where L_{CE}, θ, y_i, x_i, and $p(.)$ denote the CE loss, parameters of the DNN, the class label of the ith input feature vector, and the a posteriori output at the DNN output layer, respectively.

3.2. Joint-Softmax-Center [17]

This loss function is introduced in [17] to develop robust discriminative deep features considering two loss functions together in training DNNs for face recognition. To investigate the effectiveness of this loss function for TD-SV, we train a feed-forward DNN with joint supervision of softmax L_s and center loss L_c functions for extracting BN features such as:

$$L_{jsc} = L_s + \lambda L_c \tag{4}$$

$$= -\sum_{i=1}^{N} \log \frac{e^{W'_{y_i} z_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W'_j z_i + b_j}} + \frac{\lambda}{2} \sum_{i=1}^{N} \|z_i - c_{y_i}\|^2 \tag{5}$$

where $z_i \in R^d$ denotes the ith d dimensional deep feature belonging to the y_i class. $W_j \in R^d$ and $b \in R^n$ denote the jth column of the weight matrix $W \in R^{d \times n}$ in the last layer of DNN and the bias, respectively. N and n denote the number of samples in a mini-batch and the number of classes, respectively. $c_{y_i} \in R^d$ denotes the centroid of the y_i class in deep feature space. c_{y_i} is updated over each mini-batch, and L_c characterizes the intra-class

variation. $(.)'$ denotes the transpose operation. We consider $d = 128$ (the embedding feature dimension, i.e., the dimension of the last DNN layer) and the balancing factor λ for two losses to be 0.003 (as per [17]).

3.3. Modified Softmax [18]

It is observed in [18] that the learned feature with softmax exhibits an angular distribution, and hence, the combination of different euclidean distance-based loss functions (triplet loss [22] and contrastive loss [48])) may not be well suited with softmax. Therefore, the softmax function with the angular margin is introduced in [18] for face recognition, and the learned feature with this loss function will be angularly distributed. In our work, a feed-forward DNN is trained to discriminate between the speakers at the output layer with a modified softmax-based cross-entropy function L_{ms} such as:

$$L_{ms} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\|z_i\|\cos(\theta_{y_i,i})}}{\sum_j e^{\|z_i\|\cos(\theta_{j,i})}} \tag{6}$$

where $\theta_{j,i}$ $(0 \leq \theta_{j,i} \leq \pi)$ denotes the angle between the d dimensional deep feature (or embedding) z_i (of the ith sample belonging to the y_ith class) and weight vector W_j (the jth column of weight matrix $W \in R^{d\times n}$). n denotes the number of classes. θ_{y_i} defines the angle between the learned feature z_i and the weight vector W_{y_i} (the y_ith column of W). We consider the embedding feature dimension (i.e., the dimension of the last layer of DNN) $d = 128$. For more details, see [18].

3.4. ArcFace [19]

This loss function is introduced in [19] to improve the discrimination capability of a face recognition model by adding an angular penalty margin on the embedding features in the hyper-plane. The discrimination is obtained by increasing and decreasing the inter- and intra-class dispersion, respectively. It is shown in [19] that ArcFace yields better accuracy in face recognition than the existing 10 benchmark methods such as triplet-loss, softmax-loss, and center-loss. The ArcFace loss function is defined as

$$L_{arc} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m)}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n} e^{s\cos\theta_j}} \tag{7}$$

where θ_j defines the angle between the weight vector W_j (the jth column vector of weight matrix $W \in R^{d\times n}$) and the deep feature vector $z_i \in R^d$ (of the ith sample belonging to the y_ith class). θ_{y_i} defines the angle between the feature z_i (of class y_i) and weight vector W_{y_i}. d denotes the dimension of the embedded deep feature of the ith sample of class y_i. N and n denote the batch size and the number of classes, respectively. m adds the angular margin penalty between the z_i and W_{y_i} to increase the compactness and discrepancy for the intra-class and inter-class, respectively. s is a scaling factor. The angle θ_j, feature z_i, and weight vector W_j are related as

$$W_j^t z_i = \|W_j^t\|\|z_i\|\cos\theta_j. \tag{8}$$

In our experiments, the dimension of the DNN output layer, i.e., the value of d, is set to 128. For more details, see [19].

3.5. Focal [20]

This loss function is proposed in [20], especially for object detection in imbalance class scenarios, which basically downgrades the importance of the easily classified examples to avoid being overwhelmingly dominated by the easy negative examples in the model training. This system is analogous to the *BN-spkr* with cross-entropy loss. The only

difference is that it incorporates a modulating factor $(1-p_t)^{\gamma}$ with the cross-entropy-based loss function. It can be expressed as

$$L_{focal} = -(1-p_t)^{\gamma}\log(p_t) \tag{9}$$

where $\gamma \in [0,5]$. For $\gamma = 0$, Equation (9) becomes the equivalent to cross-entropy-based loss function, and a high value of γ increases the effect of the modulating factor. p_t denotes the posterior probability of the target class estimated by the model. More details can be found in [20]. For the well-classified case of sample t, $p_t \to 1$ and the modulating factor becomes 0, and thus, the loss is down-weighted for the well-classified example. The value of γ is considered 2, as in [20]. In our experiments, the number of speech samples and their duration vary across speakers, so it represents an imbalanced class scenario.

3.6. OSL [21]

To reduce the over-fitting problem of DNN trained with a small training set, the inclusion of an orthogonal softmax layer in classification is proposed in [21] for scene classification. It maximizes the classification margin by increasing the angle among the weight vectors of different classes. In this method, an orthogonal softmax layer is defined at the output layer of DNN as

$$r = softmax((\omega * W)\psi) \tag{10}$$

where $*$ represents the element-wise product and ω indicates the predefined fixed block diagonal mask matrix. OSL makes orthogonal the weight vectors in the classification layer during both the training and test processes, which leads to a tighter generalization error bound. ψ and r stand for the input and output vectors of the layer, respectively.

3.7. Triplet-Loss [22]

This loss function is proposed in [22] for embedding a face image into a low dimensional space with the purpose of discriminating the positive examples from the negative ones based on a distance margin. This method achieves very high accuracy in face recognition. To use the loss function for BN feature extraction in TD-SV, a feed-forward DNN is trained to discriminate speakers with a loss function that minimizes the distance between the anchor and the positive class and maximizes the distance between the anchor and the negative class. It can be expressed as

$$L_{triplet} = max\big(max[d(z_a, z_p)] - min[d(z_a, z_n)] + \ margin, 0\big) \tag{11}$$

where z_a, z_p, and z_n represent the anchor, positive, and negative embeddings, respectively. For the distance measurement $d(.,.)$ in Equation (11), the input feature vectors of training speakers are embedded into the 128 dimensional vector space at the last layer of DNN. The triplet score is calculated on the embedded space, i.e., at the last layer of DNN. We consider online triplet loss, i.e., an example within the same class as the anchor is considered to be positive and an example from different classes than that of the anchor is considered to be negative within the data samples of a particular mini-batch. Afterward, the frame level output from a particular hidden layer of DNN for a given utterance is projected onto a low dimensional space to obtain the BN feature.

3.8. SimCLR [23]

The SimCLR is proposed in [23] for the useful visual representation in image classification. It yields the best results in top-1 accuracy compared to other methods in the ImageNet dataset. The SimCLR function $L_{CLR}(i,j)$ for a pair of examples within the positive class (same class) is defined as

$$\begin{aligned} L_{CLR}(i,j) &= -\log \frac{exp(sim(z_i,z_j)/\tau)}{\sum_{k=1}^{2N} 1_{[k\neq i]} \quad \exp(sim(z_i,z_j)/\tau)} \\ &= -sim(z_i,z_j)/\tau \\ &\quad + \log \sum_{k=1}^{2N} 1_{[k\neq i]} \quad \exp(sim(z_i,z_j)/\tau) \end{aligned} \tag{12}$$

where $sim(z_i,z_j) = \frac{z_i^t z_j}{\|z_i\|\|z_j\|}$, and $1_{[k\neq i]}$ indicates 1 iff $i \neq k$. τ is called the temperature parameter. z_i denotes the d dimensional embedded deep feature for input sample x_i. We consider $d = 128$, i.e., the dimension of the DNN output/embedding layer. The final loss is computed over all positive pairs available, i.e., both (i,j) and (j,i) in the particular mini-batch data. For more details, see [23].

4. Activation Functions

In this section, we describe the different activation functions that are broadly used in many fields, including speech processing.

4.1. Sigmoid [24]

This is a non-linear activation function, defined as

$$f_{sgm}(v) = \frac{1}{1+e^{-v}} \tag{13}$$

$$\frac{df_{sgm}(v)}{dv} = f_{sgm}(v)(1-f_{sgm}(v)) \tag{14}$$

$$= \frac{e^{-v}}{(1+e^{-v})^2} \to 0, \text{if } v \to \pm \text{ large-value} \tag{15}$$

where v is the input to the activation function. As in Equation (13), the sigmoid function squishes its input to a value between 0 to 1, and hence, the large change in the input yields a small change in output (with the maximum value of 1), as shown in Figure 2. Therefore, the parameter optimization of a DNN through error back-propagation faces the known gradient vanishing problem. Specifically, the multiplication of the gradient with a small value (as Equation (15) shows) across different layers in deep networks during the back-propagation process yields an exponential decaying of the gradient. As a result, the weights and biases of the initial layers are not updated sufficiently during the training process. Nevertheless, this function is widely used in speaker and language recognition.

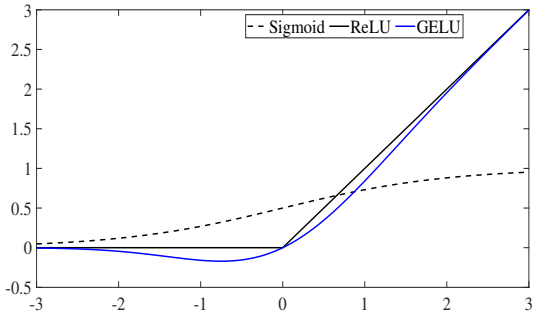

Figure 2. The sigmoid, ReLU, and GELU activation functions.

4.2. ReLU [26]

ReLU is a piece-wise linear activation function defined as

$$f_{ReLU}(v) = max(0,v) = \begin{cases} v, & \text{if } v \geq 0 \\ 0, & \text{if } v < 0 \end{cases} \tag{16}$$

ReLU preserves the dynamic range of the input in the output when the input is greater than zero, as shown in Equation (16) and Figure 2. Therefore, it does not suffer from the gradient vanishing problem as the sigmoid function does. Additionally, it provides better and faster convergence [38] compared to the sigmoid function, which makes it very popular in state-of-the-art DNN systems with a variety of applications [49]. However, it is not statistically motivated.

4.3. Leaky ReLU [50]

The Leaky ReLU activation function introduces a slope α into ReLU and is defined as,

$$f_{LeakyReLU}(v) = \begin{cases} v, & \text{if } v > 0 \\ \alpha v, & \text{if } v \leq 0 \end{cases} \tag{17}$$

Leaky ReLU is similar to ReLU, except for the negative slopes when input ≤ 0, and it is helpful in a situation when a large number of neurons are dead (i.e., no gradient flows) in the network.

4.4. GELU [38]

As discussed above, the sigmoid function suffers from the gradient vanishing problem and the ReLU function is statistically less motivated. To tackle the problem of the lack in the probabilistic interpretation of ReLU, stochastic regularization, e.g., dropout, is often introduced to improve the training of DNNs. In an attempt to merge the probabilistic regularization with an activation function, GELU is proposed. It is a standard Gaussian cumulative distribution function that introduces the non-linearity onto the output of a DNN neuron based on their values, instead of using the input sign as in ReLU. GELU is defined as

$$\begin{aligned} f_{GELU}(v) &= vp(V \leq v) && (18)\\ &= v\phi(v) && (19)\\ &= 0.5v\left(1 + erf\left(\frac{v}{\sqrt{2}}\right)\right) && (20) \end{aligned}$$

where v and $\phi(v)$ are the input to the activation function and cumulative distribution function $\mathcal{N}(0,1)$, respectively. Figure 2 illustrates the sigmoid, ReLU, and GELU activation functions.

5. Classifiers

In this section, we describe the different modeling techniques that are commonly used in speaker verification.

5.1. GMM-UBM

In this method [41], a GMM-UBM is trained using data from many non-target speakers. Then, the target speaker models are obtained from the GMM-UBM, λ_{ubm}, with maximum a posteriori (MAP) adaptation in the enrollment phase. During the test, the feature vector of the test utterance $X = \{x_1, x_2, \ldots, x_N\}$ is scored against the claimant λ_{tar} and GMM-UBM models. Afterward, the log-likelihood ratio (LLR) value is calculated for decision-making:

$$LLR(X) = \frac{1}{N}\sum_{i=1}^{N}\{\log\ p(x_i|\lambda_{tar}) - \log\ p(x_i|\lambda_{ubm})\} \tag{21}$$

Figure 3 illustrates a text-dependent speaker verification system using the GMM-UBM technique. No labeled data are required for training GMM-UBM.

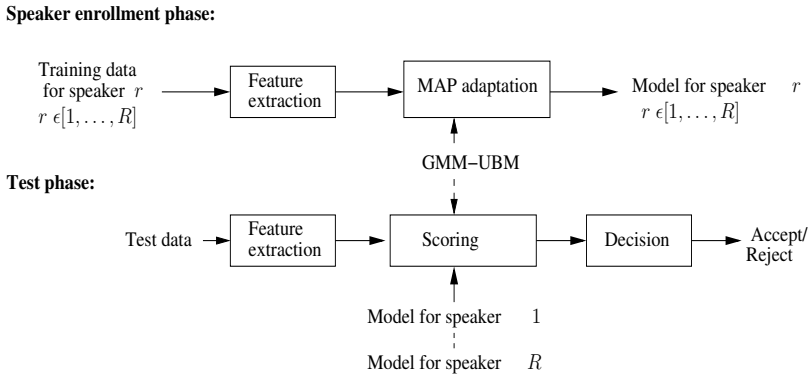

Figure 3. Text-dependent speaker verification using GMM-UBM.

5.2. i-Vector with PLDA

In this method [2], a speech signal is represented using a low-dimensional vector called i-vector, which is obtained by projecting the signal into a low dimensional subspace (called the total variability (T) space) of a speaker-independent GMM-UBM super-vector, where the speaker and channel information is assumed to be dense. For a given speech signal of a speaker, the speaker and channel-dependent GMM super-vector S can be expressed as

$$S = M + T\omega \tag{22}$$

where M denotes the speaker-independent GMM super-vector, and ω is called an i-vector. During the enrollment phase, each speaker is represented by an average i-vector computed over his/her training utterance-wise (or speech session-wise) i-vectors. In the test phase, the i-vector of a test utterance ω_t is scored against the claimant-specific i-vector ω_e (obtained during enrolment) with PLDA [4]. Figure 4 illustrates TD-SV using the i-vector technique.

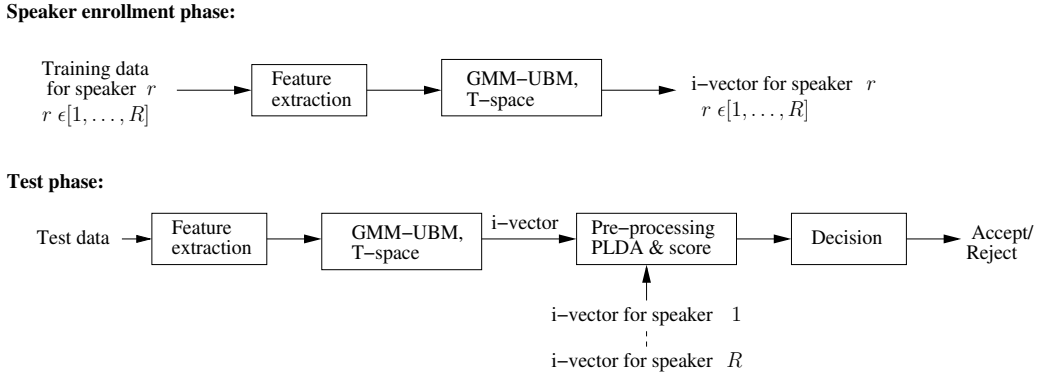

Figure 4. Text-dependent speaker verification using i-vector.

6. Experimental Setup

For evaluation, male speakers of the m-part-01 task in the RedDots challenge 2016 database are used as per protocol [51] and the database (is composed of 35 target males, 14 unseen male imposters, 6 target females, and 7 unseen female speakers). The task consists of 320 target models (from 35 target male speakers) for training using the recording of three voice samples for a particular pass-phrase. Each utterance is very short in duration, an average of 2–3 s. Three types of non-target trials are available for the performance of the TD-SV system:

- Target-wrong (TW): When a genuine speaker speaks a wrong phrase, i.e., a different pass-phrase/sentence in testing compared to their enrollment phrase.
- Imposter-correct (IC): When an imposter speaks a sentence/pass-phrase in testing where the pass-phrase is the same as that of the target enrollment sessions.

- Imposter-wrong (IW): When an imposter speaks a sentence/pass-phrase to access the system where the pass-phrase is different from that of the target enrollment sessions.

The evaluation data set is further divided into a development set (devset) and a test set (called the evaluation-set interchangeably) as per [52–54]. The development set consists of a disjoint set of nine speakers (who are excluded from the system evaluation) and the rest for evaluation. Finally, it yields 72 and 248 target models for development and evaluation, respectively. It is important to note that the trials in the devset are derived by cross-claiming of one speaker against the others (within the nine speakers). However, the evaluation set consists of some imposter trials that are from speakers outside the enrollment speakers, i.e., unknown, and this makes the evaluation set more challenging than the devset and useful for real-world scenarios where the system can encounter unknown imposters. Table 1 shows the number of different trials available in the development and evaluation sets. For more details about the database, see [51].

Table 1. Number of trials available for the development and evaluation sets.

Data Set	# of True Trials	# of Trials in Non-Target Type		
		Target -Wrong	Imposter -Correct	Imposter -Wrong
Development	1123	10,107	8013	72,125
Evaluation	2119	19,071	62,008	557,882

For the spectral feature, 57 dimensional MFCC feature vectors (19 static and their first and second derivatives) are extracted from speech samples with RASTA filtering [55] using a 25 ms hamming window and a 10 ms frame shift. After extracting the features, rVAD [56], an open-source unsupervised voice activity detection (VAD) algorithm (https://github.com/zhenghuatan/rVAD, accessed on 16 March 2022) is applied to discard the low energized frames. Finally, the selected frames are normalized to zero mean and unit variance at the utterance level.

In the GMM-UBM system, the GMM-UBM of 512 mixtures (having diagonal covariance matrices) is trained using 6300 speech files from the TIMIT database [57] with over 438 males and 192 females. Three iterations of MAP adaptation are considered during the training of the speaker-dependent model with the value of relevance factor 10. For training DNNs for BN feature extraction and training total variability and PLDA for i-vector systems, 72,764 utterances over 27 pass-phrases (of 157 male and 143 female speakers) from the RSR2015 database [58] are used.

For BN feature extraction, DNNs with six hidden layers are trained with the following configuration: a batch size of 1024, learning rate of 0.001, 30 training epochs, 1024 neurons per hidden layer, and the contextual input of 11 frames (i.e., 5 left frames, 1 current frame, and 5 right frames). The number of target speakers in BN-spkr is 300. BN features are extracted by projecting the frame level output for a particular hidden layer (before applying the activation function) of DNNs onto 57 dimensional space using PCA to align with the dimension of the MFCC feature for a fair comparison.

TensorFlow [59] is used for training the DNNs for all BN features, except for APC-BN. The examples from the same class within a mini-batch are considered as positive, and the examples from classes other than a particular positive class are treated as negative for similarity measures for those loss functions (triplet-loss and SimCLR) that require positive and negative examples. The process is repeated for all samples within the mini-batch. The values of s, m, and τ are considered, respectively, 64, 0.5, and 0.5 in both Archface and SimCLR. L_2 regularization is considered during the training of DNNs with a penalty value of 0.0001. In Leaky ReLU, the value of the slope parameter is considered to be 0.1.

For extracting APC-BN features, the DNN encoder is trained as per [15], which consists of 3 hidden layers in the gated recurrent unit (GRU) with the following configuration: a

batch size of 32, a learning rate of 0.001, and $t_n = 5$, as in Equation (2) (which gives the best performance in [15]).

In PLDA, speaker and channel factors are kept full, and the same pass-phrase utterances from a particular speaker are considered as an individual speaker. It gives 8100 classes (4239 males and 3861 females). The i-vector system is implemented using the Kaldi toolkit [60]. PCA is trained by the data set used for training the GMM-UBM.

System performance is measured in terms of equal error rate (EER) and minimum detection cost function (minDCF), as per the 2008 SRE [61]. Note that our discussions on experimental results will be primarily centered around EER to be concise as EER and minDCE results mostly agree with each other. The detection cost function is defined as

$$DCF = C_{Miss} \times P_{Miss|Target} \times P_{Target} + C_{FA} \times P_{FA|NonTarget} \times (1 - P_{Target}) \quad (23)$$

where $C_{Miss} = 10$, $C_{FA} = 1$, and $P_{Target} = 0.01$.

7. Results and Discussions

This section presents experimental results using the methods presented above and analyzes the results.

7.1. Performance of Spkr-BN Features

In Table 2, we present the TD-SV performance of Spkr-BN features using different activation functions, different loss functions, and different DNN hidden layers on the development and evaluation sets using the GMM-UBM technique for SV. For simplicity, the average EER and MinDCF values across TW, IC, and IW non-target trials are included. The TD-SV performance of each BN feature is represented by its performance on the evaluation set, for which the particular hidden layer performing the best (giving the lowest average EER) on the development set is chosen. The same hidden layer (i.e., the best-performing layer for GMM-UBM) is used for evaluating the i-vector technique.

First, we compare the performance of different activation functions. From Table 2, it is noticed that GELU-based BN features give, in most cases, the lowest average EER values compared with sigmoid and ReLU. More specifically, the widely used sigmoid function in general performs significantly worse, and the performance difference between GELU and RELU is small. This demonstrates the superiority of GELU as the activation function for DNN-based BN feature extraction in TD-SV. As the ReLU function is broadly used and the leaky ReLU function is a slightly modified version of ReLU, we extensively studied ReLU first and then conducted extended experiments on leaky ReLU.

Then, we compare the different loss functions. It is seen that CE, joint-softmax-center, and focal show the overall lowest average EER values, and they are largely on par. They are followed by ArchFace and OSL loss functions. Triplet and SimCLR loss functions perform the worst in these experiments, and when these two loss functions are applied, the impact of choosing different activation functions is negligible. This could be due to the fact that they require special care in selecting or even generating negative and positive examples (considering SimCLR is a self-supervised learning approach) [23,62].

Now, we look at the TD-SV performance of BN features using different hidden layers on devset, as shown in Table 2. We can see that for ReLU and GELU, the early hidden layer-based BN features, in general, perform better. Interestingly, it is observed that when the BN features are extracted with hidden layers close to the output of the DNN, the sigmoid-based features yield lower error rates than those using ReLU and GELU. This could be explained by the fact that the sigmoid function suffers from the vanishing gradient problem, and thus, the training focuses more on the later layers than the initial layers.

Table 2. TD-SV performance (average EER/MinDCF) of Spkr-BN features using different loss functions and hidden layers on the development and evaluation sets using the GMM-UBM technique. The performance on the evaluation set is based on the particular hidden layer that performs the best on the development set.

(**a**) Cross-Entropy							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.23/1.10	2.93/1.04	2.90/1.10	2.84/1.03	2.61/1.01	**2.57**/1.07	2.06/0.87
ReLU	**1.94**/0.65	2.10/0.70	2.22/0.76	2.67/0.90	3.67/1.33	5.53/1.84	1.28/0.51
GELU	**1.86**/0.67	1.91/0.67	2.23/0.84	2.87/1.01	3.97/1.35	5.71/1.92	**1.26**/0.49
(**b**) Joint-softmax-center							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.07/1.02	2.71/0.96	2.65/0.97	2.43/1.03	2.96/1.08	**2.34**/1.02	1.99/0.85
ReLU	**1.99**/0.62	2.22/0.68	2.37/0.81	2.34/0.84	3.29/1.12	5.36/1.63	1.35/0.49
GELU	**1.77**/0.66	2.05/0.71	2.04/0.80	2.56/0.90	3.37/1.22	5.05/1.75	**1.25**/0.51
(**c**) Modified Softmax							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.37/1.07	3.02/1.12	3.32/1.22	**2.85**/1.07	3.04/1.10	3.06/1.12	2.15/0.81
ReLU	2.08/0.77	**1.86**/0.71	2.28/0.74	2.94/1.04	3.90/1.35	6.64/2.38	**1.40**/0.59
GELU	1.96/0.79	**1.77**/0.78	2.13/0.82	2.66/1.02	3.33/1.28	4.70/1.65	1.54/0.62
(**d**) ArchFace							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.29/1.15	2.92/1.09	**2.55**/1.01	2.66/1.03	2.83/1.15	3.04/1.14	2.17/0.82
ReLU	**1.96**/0.69	2.28/0.74	2.40/0.88	3.45/1.16	4.95/1.68	6.98/2.56	1.45/0.52
GELU	**1.83**/0.66	2.23/0.74	2.20/0.76	2.59/1.01	4.19/1.54	6.64/2.17	**1.37**/0.54
(**e**) Focal							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	2.82/1.12	3.09/1.12	3.14/1.13	2.67/1.08	**2.61**/1.03	2.76/1.02	2.04/0.82
ReLU	**1.86**/0.67	1.89/0.69	2.30/0.86	2.77/0.89	3.91/1.32	4.82/1.61	1.37/0.51
GELU	**1.80**/0.66	1.87/0.66	2.46/0.83	2.58/0.90	4.23/1.38	5.37/1.83	**1.29**/0.51
(**f**) Focal							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	2.75/1.03	3.16/1.04	2.92/1.05	3.24/1.01	**2.64**/0.99	2.91/1.04	2.24/0.86
ReLU	**2.05**/0.69	2.13/0.76	3.09/1.01	3.78/1.32	5.69/2.05	8.06/2.52	**1.33**/0.53
GELU	**1.93**/0.71	2.71/0.79	2.72/0.90	3.82/1.27	5.25/1.81	7.41/2.24	1.43/0.50
(**g**) Triplet (Cosine)							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	2.69/1.07	2.83/1.01	**2.57**/1.00	2.83/1.08	2.85/1.11	2.71/1.04	2.34/0.85
ReLU	3.26/1.11	**3.17**/1.19	3.45/1.24	3.64/1.30	4.37/1.60	4.95/1.82	**2.31**/0.90
GELU	3.07/1.11	**2.82**/1.05	3.29/1.21	3.08/1.36	5.10/1.87	9.24/2.89	2.38/0.89

Table 2. *Cont.*

(**h**) Triplet (Euclidean)							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.05/1.11	2.89/1.06	3.17/1.11	**2.79**/1.07	3.11/1.17	2.83/1.08	**2.17**/0.79
ReLU	**2.91**/1.13	3.20/1.11	3.03/1.17	3.62/1.36	4.50/1.66	5.18/2.04	2.21/0.83
GELU	**2.79**/1.08	3.14/1.20	3.64/1.28	4.42/1.55	6.49/2.05	14.11/3.61	2.21/0.84
(**i**) SimCLR							
Activation function	Development-set (Hidden Layer)						Evaluation-set
	Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
Sigmoid	3.30/1.13	2.93/1.07	3.53/1.17	**2.85**/1.01	3.20/1.11	3.00/1.06	2.22/0.86
ReLU	3.09/1.07	2.87/1.06	**2.72**/1.18	3.40/1.35	3.89/1.53	4.96/1.85	2.51/1.03
GELU	**3.11**/1.11	3.46/1.12	3.23/1.26	3.72/1.29	5.28/1.71	7.95/2.56	**2.09**/0.80

7.2. Performance of TCL-BN Features

In Table 3, we compare the performance of TCL-BN features with the cross-entropy loss function but with different activation functions and different hidden layers using the GMM-UBM technique for SV. It can be seen that the uTCL-BN method outperforms sTCL-BN, which is in line with [1]. It is shown in [1] that uTCL (using the sigmoid activation function) is very competitive and superior to the compared methods. In this work, we can observe that uTCL using the GELU activation function further reduces the error rate compared to the sigmoid and ReLU activation functions.

Table 3. TD-SV performance (average EER/MinDCF) of uTCL-BN features using different activation functions and different hidden layers on the development and evaluation sets using the GMM-UBM technique. The loss function is cross entropy.

Feature	**Activation Function**	**Development-Set (Hidden Layer)**						**Evaluation-Set**
		Ly1	**Ly2**	**Ly3**	**Ly4**	**Ly5**	**Ly6**	
uTCL-BN	Sigmoid	**1.98**/0.79	2.01/0.78	2.55/0.86	4.03/1.64	7.55/3.24	27.39/7.59	1.38/0.55 (Ly1)
	ReLU	1.72/0.59	1.53/0.61	1.58/0.59	**1.46**/0.63	1.92/0.72	1.99/0.76	1.40/0.59 (Ly4)
	GELU	1.80/0.61	**1.50**/0.58	1.54/0.60	1.59/0.65	2.20/0.74	2.46/0.86	**1.08**/0.45 (Ly2)
sTCL-BN	Sigmoid	3.35/1.10	**2.84**/1.06	3.11/1.10	2.72/1.03	3.05/1.06	2.94/1.04	**2.23**/0.82 (Ly2)
	ReLU	3.08/1.08	**2.79**/1.10	3.17/1.20	4.01/1.48	5.94/2.23	33.90/9.67	2.44/0.84 (Ly2)
	GELU	**2.70**/1.10	3.07/1.10	3.47/1.22	3.41/1.26	3.85/1.24	3.56/1.31	2.25/0.83 (Ly1)

Furthermore, uTCL-BN with GELU (with EER of 1.08%) also outperforms, by a large margin, the best-performing Spkr-BN feature, which is based on cross-entropy (with EER of 1.26%) or the joint-softmax-center (with EER of 1.25%) with GELU as well.

To further investigate the reason why GELU-based BN features yield much lower EER in TD-SV than sigmoid, we scatter-plot Spkr-BN and uTCL-BN features for different activation functions using T-SNE [63] with the same parameters, as shown in Figure 5. The figure depicts that GELU-based features demonstrate more discriminative patterns than sigmoid-based ones and MFCCs, which is also reflected by the EER values of the corresponding features (as shown in Tables 2–4).

As SV is fundamentally a classification problem, the more discriminative feature is expected to yield a better separability between classes in the score domain. Therefore, we plot in Figure 6 the LLR score distributions of target-true (genuine) and impostor-correct (impostor) trials of the Spkr-BN-based GMM-UBM systems on the evaluation set (see Table 2) with sigmoid and GELU activation functions for the layers *Ly*6 and *Ly*1, respectively (which yields the lowest error rate on the development set for the respective activation functions on the same system), to demonstrate the impact of different activation

functions. The figure shows that the GELU-based system yields mostly higher scores for the target-true and lower scores for imposter-correct trials compared to the sigmoid-based system. This further indicates that GELU is a better choice.

Table 4. TD-SV performance (average EER/MinDCF) of APC-BN features using different activation functions and different hidden layers on the development and evaluation sets using the GMM-UBM technique. The loss function is $\ell 1$. Ly{1,3} denotes the concatenation of outputs from hidden layers 1 and 3.

Feature	Activation Function	Development-Set (Hidden Layer)							Evaluation-Set	
		Ly1	Ly2	Ly3	Ly{1,2}	Ly{1,3}	Ly{2,3}	Ly{1,2,3}	Ly2	Ly{1,3}
MFCC	-	-	-	-	-	-	-	-	2.23/0.84	
APC-BN	Sigmoid	2.79/0.95	1.99/0.73	2.46/0.88	2.29/0.76	**1.99**/0.74	2.08/0.86	**1.99**/0.66	1.21/0.53	1.26/0.53
	ReLU	2.46/0.98	2.10/0.74	2.34/0.88	2.20/0.74	**1.99**/0.74	2.21/0.78	2.00/0.69	1.30/0.58	1.27/0.51
	GELU	2.38/0.93	1.89/0.72	2.63/0.98	1.97/0.71	**1.83**/0.64	2.08/0.77	2.05/0.75	1.22/0.54	**1.18/0.48**

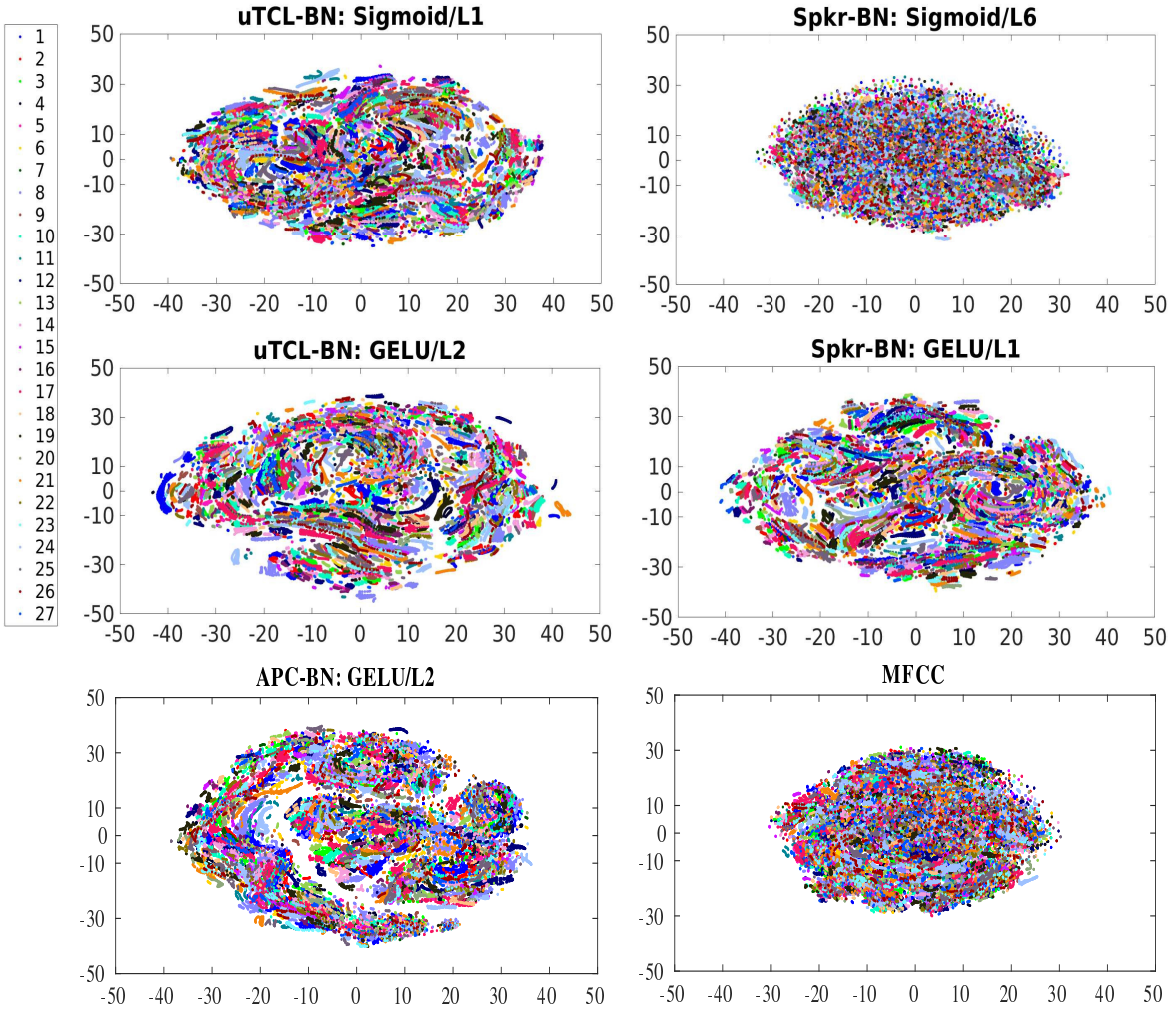

Figure 5. Scatter plots of MFCCs and BN features extracted for the target speakers whose utterances are available in the evaluation set, using T-SNE [63] with the same parameters. All features are extracted from the same utterances for a fair comparison.

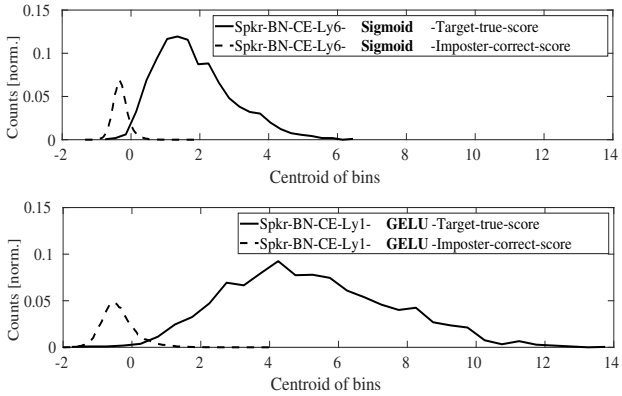

Figure 6. Distribution of the target-true and imposter-correct scores of the GMM-UBM TD-SV system in the evaluation set for Spkr-BN with sigmoid and GELU activation functions. All systems use the same trials for a fair comparison.

7.3. Performance of APC-BN Features

In Table 4, we present the TD-SV performance of APC-BN features using different activation functions and different hidden layers on the development and evaluation sets using the GMM-UBM technique. From Table 4, it can be observed that GELU in general outperforms sigmoid and ReLU, and they all are significantly superior to MFCC.

In addition, the concatenation of APC-BN features extracted from different hidden layers further slightly reduces the average EER and minDCF values. This indicates that different layers of an APC network capture different speaker-related information, and hence, it is beneficial to combine them. Note that we also performed the experiments by concatenating features extracted from different hidden layers for uTCL-BN or Spkr-BN, but none of the combinations yielded any gain and, thus, were not shown in the paper.

7.4. Overall Comparison and Score Fusion

Table 5 further compares the TD-SV performance of BN features extracted using the Leaky ReLU activation function with those winning configurations (low average error rates) in Tables 2a, 3, and 4. It can be observed that Leaky ReLU is very competitive with GeLU. However, GeLU provides lower error rates in most cases, and hence, GeLU remains the best among the studied functions for BN feature extraction in TD-SV based on uTCL and APC-BN. In Table 6, we first summarize and compare the results across three different types of BN features: Spkr-BN in Table 2, uTCL-BN in Table 3, and APC-BN in Table 4 by picking up the best-performing configuration from each category. We can see (1) all BN features outperform MFCCs significantly; (2) uTCL-BN performs the best, followed by APC-BN, which both use self-supervised training targets; and (3) GELU is the best-performing activation function across all three training targets. Table 6 further presents the detailed performance for each of the three non-target type trials. An interesting observation from the table is that both APC-BN and uTCL-BN show a large reduction in EER for the target-wrong and imposter-wrong trials compared to Spkr-BN, while Spkr-BN performs better for imposter-correct trials. It indicates that APC-BN and uTCL-BN are better at modeling the temporal or phonetic information available in the speech signal in a self-supervised manner, which benefits TD-SV. It should be noted that there are a variety of supervised and self-supervised training targets available in the literature, and we select a few typical examples only in this work with no intention to make exhaustive comparisons in this spectrum. Furthermore, the simple score fusion (averaging scores with equal importance) of the three systems selected from each category brings further performance improvement over their standalone counterparts. This indicates that these features carry information complementary to each other.

Table 5. TD-SV performance (average EER/MinDCF) of spkr-BN, uTCL-BN, and APC-BN features using the Leaky ReLU activation function compared to those winning configurations (low average error rates) in Tables 2a, 3, and 4 for the respective systems.

Feature	Activation Function	Development-Set (Hidden Layer)						Evaluation-Set
		Ly1	Ly2	Ly3	Ly4	Ly5	Ly6	
MFCC	-	-	-	-	-	-	-	2.23/0.84
Spkr-BN	ReLU	**1.94**/0.65	2.10/0.70	2.22/0.76	2.67/0.90	3.67/1.33	5.53/1.84	1.28/0.51 (Ly1)
	Leaky ReLU	2.17/0.69	**1.92**/0.68	2.074/0.73	2.49/0.89	3.05/1.13	4.27/1.57	1.37/0.55 (Ly2)
	GELU	**1.86**/0.67	1.91/0.67	2.23/0.84	2.87/1.01	3.97/1.35	5.71/1.92	**1.26**/0.49 (Ly1)
uTCL-BN	ReLU	1.72/0.59	1.53/0.61	1.58/0.59	**1.46**/0.63	1.92/0.72	1.99/0.76	1.40/0.59 (Ly4)
	Leaky ReLU	1.74/0.65	1.54/0.52	**1.53**/0.60	1.81/0.65	2.08/0.76	2.25/0.86	1.25/0.51 (Ly3)
	GELU	1.80/0.61	**1.50**/0.58	1.54/0.60	1.59/0.65	2.20/0.74	2.46/0.86	**1.08**/0.45 (Ly2)
		Ly1	Ly2	Ly3	Ly{1,2}	Ly{1,3}	Ly{2,3}	
APC-BN	ReLU	2.46/0.98	2.10/0.74	2.34/0.88	2.20/0.74	**1.99**/0.74	2.21/0.78	1.27/0.51 (Ly{1,3})
	Leaky ReLU	2.58/0.95	**1.84**/0.74	2.52/0.90	2.22/0.82	1.85/0.66	1.89/0.72	**1.18**/0.50 (Ly2)
	GELU	2.38/0.93	1.89/0.72	2.63/0.98	1.97/0.71	**1.83**/0.64	2.08/0.77	**1.18/0.48** (Ly{1,3})

Table 6. TD-SV performance for the different types of non-target trials for different combinations of activation functions and loss functions on the evaluation set using the GMM-UBM technique.

Feature	Loss Function	Activation Function /Hidden Layer	Non-Target Types [%EER/MinDCF × 100]			Avg. EER/ MinDCF
			Target-Wrong	Imposter-Correct	Imposter-Wrong	
MFCC	-		3.44/1.23	2.50/1.08	0.75/0.22	2.23/0.84
Spkr-BN	CE	GELU /Ly1	1.41/0.53	**1.91**/0.87	0.47/0.09	1.26/0.49
uTCL-BN	CE	GELU /Ly2	**0.84**/0.33	2.07/0.97	**0.33**/0.07	**1.08/0.45**
APC-BN	$\ell 1$	GELU /Ly{1,3}	1.03/0.34	2.12/1.03	0.38/0.08	1.18/0.48
Score fusion Spkr + uTCL + APC-Ly{1,3} [BN]	-	GELU	**0.71**/0.28	**1.70**/0.83	**0.33**/0.06	**0.91/0.39**

7.5. TD-SV Performance of BN Features with the i-Vector Technique

Table 7 compares the performance of TD-SV with the evaluation set using the i-vector technique for those features seen in Table 6. From Table 7, it is observed that the i-vector technique exhibits similar patterns in TD-SV performance to those of the GMM-UBM systems shown in Table 6. Moreover, the score fusion drastically reduces the EER/MinDCF values with respect to their standalone counterparts. In the feature domain, simple fusion may not work [64] due to the redundancy among the features and the requirement of additional data to train the PCA for dimensionality reduction. Therefore, we will keep it for a future work.

Table 7. TD-SV performance using the i-vector technique for a number of features, presented in Table 6 on the evaluation set.

Feature	Loss Function	Activation Function /Hidden Layer	Non-Target Types [%EER/MinDCF× 100]			Avg. EER/ MinDCF
			Target-Wrong	Imposter-Correct	Imposter-Wrong	
MFCC	-	-	5.56/2.34	3.68/1.66	0.80/0.38	3.35/1.46
Spkr-BN	CE	GELU /Ly1	3.16/1.18	**3.63**/1.57	0.78/0.25	2.52/1.00
uTCL-BN	CE	GELU /Ly2	2.35/0.89	3.70/1.63	**0.60**/0.18	**2.22**/0.90
APC-BN	$\ell 1$	GELU /Ly{1,3}	**2.12**/0.69	4.20/1.89	0.61/0.16	2.31/0.91

Table 7. *Cont.*

Feature	Loss Function	Activation Function /Hidden Layer	Non-Target Types [%EER/MinDCF× 100]			Avg. EER/ MinDCF
			Target-Wrong	Imposter-Correct	Imposter-Wrong	
Fusion score Spkr + uTCL + APC (Ly{1,3}) [BN]		GELU	**1.42**/0.43	**2.64**/1.18	**0.42**/0.09	**1.49/0.57**

8. Conclusions

In this paper, we systematically studied a set of deep bottleneck (BN) feature extraction methods that are based on either supervised or self-supervised training targets for text-dependent speaker verification (TD-SV). We investigated their performance in combination with different activation functions and different loss functions in a joint framework. We further analyzed the performance when using different hidden layers for deep feature extraction. We have obtained a set of interesting results. First, all BN features outperform spectral features significantly. Secondly, the two self-supervised learning methods, utterance-wise time-contrastive learning (uTCL) and auto-regressive prediction coding (APC), both demonstrate promising and better results compared with one supervised learning approach that discriminates speaker identities. Among the three activation functions, Gaussian error linear unit (GELU) consistently and significantly outperforms sigmoid. Among a number of loss functions, cross-entropy, joint-softmax, and focal outperform the others. In the end, we show that the score-level fusion of different BN features gives further improvement. The future work will consider better fusion strategies [64] and deep neural architectures for BN feature extraction and classification to further improve the system performance.

Author Contributions: Data selection and preparation, conceptualization, methodology, software, writing the original draft manuscript, designing and running experiments, analyzing and discussing the results, A.K.S.; supervision, conceptualization, methodology, reviewing and editing the manuscript, analyzing and discussing the results, Z.-H.T. All authors have read and agreed to the published version of the manuscript.

Funding: A part of this work is supported by NLTM BHASHINI project funding 11(1)/2022-HCC(TDIL) from MeitY, Govt. of India.

Data Availability Statement: Not applicable.

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

References

1. Sarkar, A.K.; Tan, Z.-H.; Tang, H.; Shon, S.; Glass, J.R. Time-Contrastive Learning Based Deep Bottleneck Features for Text-Dependent Speaker Verification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2019**, *27*, 1267–1279. [CrossRef]
2. Dehak, N.; Kenny, P.; Dehak, R.; Ouellet, P.; Dumouchel, P. Front-end factor analysis for speaker verification. *IEEE Trans. Audio Speech Lang. Process.* **2011**, *19*, 788–798. [CrossRef]
3. Snyder, D.; Garcia-Romero, D.; Sell, G.; Povey, D.; Khudanpur, S. X-Vectors: Robust DNN Embeddings for Speaker Recognition. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April **2018**; pp. 5329–5333.
4. Senoussaoui, M.; Kenny, P.; Brümmer, N.; de Villiers, E.; Dumouchel, P. Mixture of plda models in I-vector space for gender-independent speaker recognition. In Proceedings of the Interspeech, Florence, Italy, 27–31 August **2011**; pp. 25–28.
5. Davis, S.B.; Mermelstein, P. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Trans. Acoust. Speech Signal Process.* **1980**, *28*, 357–366. [CrossRef]
6. Kim, C.; Stern, R.M. Power-Normalized Cepstral Coefficients (PNCC) for Robust Speech Recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2016**, *24*, 1315–1329. [CrossRef]
7. Hermansky, H. Perceptual Linear Predictive (PLP) Analysis of Speech. *J. Acoust. Soc. Am.* **1990**, *87*, 1738–1752. [CrossRef] [PubMed]

8. Liu, Y.; Qian, Y.; Chen, N.; Fu, T.; Zhang, Y.; Yu, K. Deep Feature For Text-dependent Speaker Verification. *Speech Commun.* **2015**, *73*, 1–13. [CrossRef]
9. Chung, Y.A.; Wu, C.C.; Shen, C.H.; Lee, H.Y.; Lee, L.S. Audio Word2Vec: Unsupervised Learning of Audio Segment Representations Using Sequence-to-Sequence Autoencoder. In Proceedings of the Interspeech, San Francisco, CA, USA, 8–12 September 2016; pp. 765–769.
10. Li, R.; Ju, C.J.T.; Chen, Z.; Mao, H.; Elibol, O.; Stolcke, A. Fusion of Embeddings Networks for Robust Combination of Text Dependent and Independent Speaker Recognition. In Proceedings of the Interspeech, Brno, Czech Republic, 30 August–3 September 2021; pp. 4593–4597.
11. Liu, Y.; Li, Z.; Li, L.; Hong, Q. Phoneme-Aware and Channel-Wise Attentive Learning for Text Dependent Speaker Verification. In Proceedings of the Interspeech, Brno, Czech Republic, 30 August–3 September 2021; pp. 101–105.
12. Mingote, V.; Miguel, A.; Ortega, A.; Lleida, E. Memory Layers with Multi-Head Attention Mechanisms for Text-Dependent Speaker Verification. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 6154–6158.
13. Du, C.; Han, B.; Wang, S.; Qian, Y.; Yu, K. SynAug: Synthesis-Based Data Augmentation for Text-Dependent Speaker Verification. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 5844–5848.
14. McLaren, M.; Lei, Y.; Ferrer, L. Advances In Deep Neural Network Approaches To Speaker Recognition. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, 19–24 April 2015; pp. 4814–4818.
15. Chung, Y.A.; Glass, J. Generative pre-training for speech with auto-regressive predictive coding. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 3497–3501.
16. Sarkar, A.K.; Tan, Z.-H. Vocal Tract Length Perturbation for Text-Dependent Speaker Verification with Autoregressive Prediction Coding. *IEEE Signal Process. Lett.* **2021**, *28*, 364–368. [CrossRef]
17. Wen, Y.; Zhang, K.; Li, Z.; Qiao, Y. A Discriminative Feature Learning Approach for Deep Face Recognition. In Proceedings of the 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Computer Vision—ECCV 2016; Volume 9911, pp. 499–515.
18. Liu, W.; Wen, Y.; Yu, Z.; Li, M.; Raj, B.; Song, L. SphereFace: Deep Hypersphere Embedding for Face Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6738–6746.
19. Deng, J.; Guo, J.; Xue, N.; Zafeiriou, S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 4685–4694.
20. Lin, T.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the of IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2999–3007.
21. Li, X.; Chang, D.; Ma, Z.; Tan, Z.H.; Xue, J.H.; Cao, J.; Yu, J.; Guo, J. OSLNet: Deep Small-Sample Classification with an Orthogonal Softmax Layer. *IEEE Trans. Image Process.* **2020**, *29*, 6482–6495. [CrossRef]
22. Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. .
23. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. *Proc. Mach. Learn. Res.* **2020**, *119*, 1597–1607.
24. Han, J.; Moraga, C. The influence of the sigmoid function parameters on the speed of backpropagation learning. In Proceedings of the From Natural to Artificial Neural Computation, IWANN, Malaga-Torremolinos, Spain, 7–9 June 1995; Lecture Notes in Computer Science; Mira, J., Sandoval, F., Eds.; Springer: Berlin/Heidelberg, Germany, 1995; Volume 930.
25. Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv* **2018**, arXiv:1811.03378.
26. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the International Conference on International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010; pp. 807–814.
27. Yaman, S.; Pelecanos, J.W.; Sarikaya, R. Bottleneck features for speaker recognition. In Proceedings of the Odyssey, Singapore, 25–28 June 2012; pp. 105–108.
28. Ghalehjegh, S.H.; Rose, R.C. Deep bottleneck features for i-vector based text-independent speaker verification. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Scottsdale, AZ, USA, 13–17 December 2015; pp. 555–560.
29. Lozano-Diez, A.; Silnova, A.; Matejka, P.; Glembek, O.; Plchot, O.; Pesan, J.; Burget, L.; Gonzalez-Rodriguez, J. Analysis and Optimization of Bottleneck Features for Speaker Recognition. In Proceedings of the Odyssey, Bilbao, Spain, 21–24 June 2016; pp. 352–357.
30. Shi, Z.; Lin, H.; Liu, L.; Liu, R. Latent Factor Analysis of Deep Bottleneck Features for Speaker Verification with Random Digit Strings. In Proceedings of the Interspeech, Hyderabad, India, 2–6 September 2018; pp. 1081–1085.
31. Fér, R.; Matějka, P.; Grézl, F.; Plchot, O.; Veselý, K.; Černocký, J. Multilingually Trained Bottleneck Features in Spoken Language Recognition. *Comput. Speech Lang.* **2017**, *2017*, 252–267. [CrossRef]

32. Ma, Z.; Yu, H.; Chen, W.; Guo, J. Short Utterance Based Speech Language Identification in Intelligent Vehicles with Time-Scale Modifications and Deep Bottleneck Features. *IEEE Trans. Veh. Technol.* **2019**, *68*, 121–128. [CrossRef]
33. Yue, Z.; Christensen, H.; Barker, J. Autoencoder Bottleneck Features with Multi-Task Optimisation for Improved Continuous Dysarthric Speech Recognition. In Proceedings of the Interspeech, Shanghai, China, 25–29 October 2020; pp. 4581–4585.
34. Ramsay, D.B.; Kilgour, K.; Roblek, D.; Sharifi, M. Low-Dimensional Bottleneck Features for On-Device Continuous Speech Recognition. In Proceedings of the Interspeech, Graz, Austria, 15–19 September 2019; pp. 3456–3459.
35. Kakouros, S.; Suni, A.; Šimko, J.; Vainio, M. Prosodic Representations of Prominence Classification Neural Networks and Autoencoders Using Bottleneck Features. In Proceedings of the Interspeech, Graz, Austria, 15–19 September 2019; pp. 1946–1950.
36. Zeiler, M.; Ranzato, M.; Monga, R.; Mao, M.; Yang, K.; Le, Q.; Nguyen, P.; Senior, A.; Vanhoucke, V.; Dean, J.; et al. On rectified linear units for speech processing. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013; pp. 3517–3521.
37. Dahl, G.E.; Sainath, T.N.; Hinton, G.E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013; pp. 8609–8613.
38. Hendrycks, D.; Gimpel, K. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *arXiv* **2016**, arXiv:1606.08415.
39. Ma, Y.; Ding, Y.; Zhao, M.; Zheng, Y.; Liu, M.; Xu, M. Poformer: A simple pooling transformer for speaker verification. *arXiv* **2021**. [CrossRef]
40. Han, B.; Chen, Z.; Liu, B.; Qian, Y. MLP-SVNET: A Multi-Layer Perceptrons Based Network for Speaker Verification. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 7522–7526.
41. Reynolds, D.A.; Quatieri, T.F.; Dunn, R.B. Speaker Verification Using Adapted Gaussian Mixture Models. *Digit. Signal Process.* **2000**, *10*, 19–41. [CrossRef]
42. Sahidullah, M.; Sarkar, A.K.; Vestman, V.; Liu, X.; Serizel, R.; Kinnunen, T.; Tan, Z.-H.; Vincent, E. UIAI System for Short-Duration Speaker Verification Challenge 2020. In Proceedings of the Spoken Language Technology (SLT) Workshop, Shenzhen, China, 19–22 January 2021.
43. Lozano-Diez, A.; Silnova, A.; Pulugundla, B.; Rohdin, J.; Veselý, K.; Burget, L.; Plchot, O.; Glembek, O.; Novotný, O.; Matejka, P. BUT Text-Dependent Speaker Verification System for SdSV Challenge 2020. In Proceedings of the Interspeech, Shanghai, China, 25–29 October 2020; pp. 761–765.
44. 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 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 4052–4056.
45. Snyder, D.; Garcia-Romero, D.; Sell, G.; McCree, A.; Povey, D.; Khudanpur, S. Speaker Recognition For Multi-speaker Conversations Using X-vectors. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 5796–5800.
46. Oord, A.V.D.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748.
47. Hsu, W.; Bolte, B.; Tsai, Y.H.; Lakhotia, K.; Salakhutdinov, R.; Mohamed, A. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *arXiv* **2021**, arXiv:2106.07447.
48. Hadsell, R.; Chopra, S.; LeCun, Y. Dimensionality Reduction by Learning an Invariant Mapping. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; Volume 2; pp. 1735–1742.
49. Ramachandran, P.; Zoph, B.; Le, Q.V. Searching for Activation Functions. *arXiv* **2017**, arXiv:1710.05941.
50. Maas, A.; Hannun, A.; Ng, A. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In Proceedings of the International Conference on Machine Learning (ICML), Altanta, GA, USA, 16–21 June 2013.
51. The RedDots Challenge: Towards Characterizing Speakers from Short Utterances. Available online: https://sites.google.com/site/thereddotsproject/reddots-challenge (accessed on 12 September 2016).
52. Kinnunen, T.; Sahidullah, M.; Kukanov, I.; Delgado, H.; Todisco, M.; Sarkar, A.K.; Thomsen, N.B.; Hautamäki, V.; Evans, N.; Tan, Z.-H. Utterance Verification for Text-Dependent Speaker Recognition: A Comparative Assessment Using the RedDots Corpus. In Proceedings of the Interspeech, San Francisco, CA, USA, 8–12 September 2016; pp. 430–434.
53. Delgado, H.; Todisco, M.; Sahidullah, M.; Sarkar, A.K.; Evans, N.; Kinnunen, T.; Tan, Z.-H. Further optimisations of constant Q cepstral processing for integrated utterance and text-dependent speaker verification. In Proceedings of the IEEE Spoken Language Technology Workshop (SLT), San Diego, CA, USA, 13–16 December 2016; pp. 179–185.
54. Sarkar, A.K.; Tan, Z.-H. Incorporating pass-phrase dependent background models for text-dependent speaker verification. *Comput. Speech Lang.* **2018**, *47*, 259–271. [CrossRef]
55. Hermanksy, H.; Morgan, N. RASTA processing of speech. *IEEE Trans. Speech Audio Process.* **1994**, *2*, 578–589.
56. Tan, Z.-H.; Sarkar, A.K.; Dehak, N. rVAD: An Unsupervised Segment-based Robust Voice Activity Detection Method. *Comput. Speech Lang.* **2020**, *59*, 1–21. [CrossRef]
57. Garofolo, J.S.; Lamel, L.F.; Fisher, W.M.; Fiscus, J.G.; Pallett, D.S.; Dahlgren, N.L.; Zue, V. *TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1*; Linguistic Data Consortium: Philadelphia, PA, USA, 1993.

58. Larcher, A.; Lee, K.A.; Ma, B.; Li, H. Text-dependent speaker verification: Classifiers, databases and RSR2015. *Speech Commun.* **2014**, *60*, 56–77. [CrossRef]
59. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: https://www.tensorflow.org/ (accessed on 1 June 2022).
60. Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P.; et al. The Kaldi Speech Recognition Toolkit. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, Big Island, HI, USA, 11–15 December 2011.
61. Available online: https://www.nist.gov/itl/iad/mig/2008-nist-speaker-recognition-evaluation-results (accessed on 1 June 2016).
62. Yu, B.; Liu, T.; Gong, M.; Ding, C.; Tao, D. Correcting the Triplet Selection Bias for Triplet Loss. In *Computer Vision—ECCV 2018, Proceedings of the 15th European Conference, Munich, Germany, 8–14 September 2018*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018. .
63. van der Maaten, L.J.P.; Hinton, G.E. Visualizing High-Dimensional Data Using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
64. Sarkar, A.K.; Do, C.T.; Le, V.B.; Barras, C. Combination of Cepstral and phonetically Discriminative Features for Speaker Verification. *IEEE Signal Process. Lett.* **2014**, *21*, 1040–1044.