IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 December 2024, accepted 22 December 2024, date of publication 30 December 2024, date of current version 6 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3523951

SURVEY

Exploring Aggregated wav2vec 2.0 Features and Dual-Stream TDNN for Efficient Spoken Dialect Identification

ANANYA ANGRA[1], H. MURALIKRISHNA[2], DILEEP AROOR DINESH [3], (Member, IEEE), AND VEENA THENKANIDIYOOR[4], (Member, IEEE)

[1]MANAS Laboratory, SCEE, Indian Institute of Technology Mandi, Mandi 175005, India
[2]Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India
[3]Department of CSE, Indian Institute of Technology Dharwad, Dharwad 580011, India
[4]Department of CSE, National Institute of Technology Goa, Ponda, Goa 403401, India

Corresponding author: H. Muralikrishna (murali.h@manipal.edu)

ABSTRACT Dialect identification (DID) is a challenging task due to high inter-class similarities between the dialects. Efficiency of a DID system depends on how well the input features encode the DID-specific contents in the speech that is spread across the utterance. In this paper, we explore different representations for efficient DID, which are motivated by the recent advancements in related areas. Firstly, we propose to learn a representation by aggregating the layers-wise features from wav2vec 2.0. We propose multiple approaches to combine the layer-wise features. Since different layers of wav2vec 2.0 are known to capture different acoustic-linguistic characteristics, such aggregated representation encode DID-specific contents in a better way. Followed by this, we explore the usage of recently proposed global-aware filter (GAF) layer based dual-stream time delay neural network (DS-TDNN) for DID. The GAF layer employs a set of learnable transform-domain filters between a 1D discrete Fourier transform and its inverse transform to capture global context along with dynamic filtering and sparse regularization. DS-TDNN has two separate input branches, one for capturing global context and the other for local context which are combined in a parallel pattern. Results obtained on dialects of Kannada, Tamil, Konkani and Marathi, four low-resource languages of India show that aggregated wav2vec 2.0 features perform better compared to DS-TDNN approach.

INDEX TERMS Spoken dialect identification, wav2vec 2.0, feature representations, DS-TDNN.

I. INTRODUCTION

Dialect represents a unique pronunciation pattern followed among a group of native people belonging to a specific geographic region. Dialectal variations in any language exist mainly due to several surrounding factors pertaining to the speaker such as, geographical location, socioeconomic status, mother tongue, the influence of neighboring state languages, cultural and education background, etc. [1], [2]. Automatic spoken dialect identification (DID) refers to the task of identifying the dialect given a predetermined language [3], [4]. DID is a more challenging task as compared to language identification (LID) because of high inter-class similarity between the dialects of a language [3]. The lack of sufficient training data further amplifies the issue. However, the deep learning models generally used for the task are data intensive.

To address the issue of drop in the performance of the DID systems due to scarcity of training data and high inter-class similarity, we intend to learn feature representations suitable for DID. Learning the features that encode dialect-specific information helps to improve the performance of the DID systems [5], [6]. One way to learn such features is to use the hand-crafted features such as mel-frequency cepstral coefficient (MFCC) [7], [8], residual mel-frequency cepstral coefficient (RMFCC) [9], etc. The other way is to use self-supervised pre-trained models such as BNF [10], wav2vec [11], wav2vec 2.0 [12], [13], etc. to extract features. The state-of-the-art (SOTA) approaches use these features as

The associate editor coordinating the review of this manuscript and approving it for publication was Orazio Gambino.

basic speech representations to train neural network based models such as fully-connected neural networks (FCNN) [7], [14], x-vector based system [7], [15], BLSTM u-vector based system [7], Transformer-based systems [16] etc. to capture dialectical cues.

Specifically, among the pre-trained feature extractors, wav2vec 2.0 generates powerful representations capturing features suitable for various speech-related tasks. It uses multi-layer convolutional feature encoder that takes as input raw audio and outputs latent speech representations for T time-steps. They are then fed to a Transformer to build representations capturing information from the entire sequence. The wav2vec 2.0 has two model configurations which use the same encoder architecture but differ in the Transformer setup, base and large [12]. The wav2vec 2.0 large model has 24 Transformer layers, whereas the wav2vec 2.0 base has 12 layers. Different layers of the wav2vec 2.0 are known to capture different characteristics present in a speech sample [17]. It is a general trend to take the output from a particular layer as the feature representation and then feed it to a FCNN to perform various speech-related tasks.

Dialects exhibit variability in pronunciation, vocabulary, grammar, and cultural references. Aggregating features from various layers allows the model to learn representations that capture different linguistic cues. The wav2vec 2.0 layers follow an autoencoder-style behavior, where as we go deeper into the model, the representation starts deviating from the input speech features followed by a reverse trend where even deeper layers become more similar to the input, as if reconstructing the input [17]. The layer-wise evolution of the representations follows an acoustic-linguistic hierarchy, where the shallowest layers encode acoustic features, followed by phonetic, word identity, and word meaning information (and then followed by a reverse trend as described above) [17], i.e., the shallowest layers does not learn the word meaning information properly. It is the middle layers that learn this information. The phonemic information is learnt by the middle layers and not by shallow layers. It is observed in the field of speaker verification and anti-spoofing that the aggregation of features by taking the weighted-sum of all hidden representations, learns the latent representations more efficiently [18], [19].

Due to their characteristics and relationships with individual speakers, many techniques used for speaker recognition are also successfully adopted for LID and DID [3]. Therefore, similar to the above approaches, we propose to integrate information from multiple layers so as to encode the diverse linguistic characteristics of different dialects, enhancing the ability of a DID system to accurately identify them. We propose to extract features from all the layers of the Transformer encoder of a pre-trained wav2vec 2.0 model. The independent representations are obtained from various layers and are combined using various pooling strategies such as statistical pooling (SP) and attentive statistical pooling (ASP) to solve the task of DID. The fusion of features helps to capture the multifaceted nature of dialectal variations, leading to improved identification performance. The feature representations of the speech utterance obtained capture DID-specific information in a better way as compared to the ones extracted from a particular layer. This is because using the aggregation across all layers helps capture information across time and layers.

Another way to learn robust feature representation suitable for DID is using recently proposed novel dual-stream time delay neural network (DS-TDNN) [20]. This architecture uses 2 parallel branches, one for capturing global and another for capturing local characteristics. The authors have used GAF based on fast-fourier transform (FFT)/inverse fast-fourier transform (IFFT) to capture long-range context [20]. Additionally, dynamic filtering (this is known as dynamic global filtering (DGF)) and sparse regularization is applied to learn generalised representations. The GAF layer can easily adapt to different audio lengths as both the FFT and the IFFT have no learnable parameters [20]. This model combines the local and global features in the parallel fashion. Therefore, DS-TDNN helps to learn powerful representations. This model is recently proposed in the field of speaker-verification. We intend to explore the feature representations obtained using DS-TDNN for dialect identification, which is a novel approach.

In the nutshell, our main contributions are as follows:

(i) Learning novel representation for DID based on statistical pooling (SP) and attentive statistical pooling (ASP) based aggregation of layer-wise features obtained from wav2vec 2.0., (ii) Learning novel representation based on DS-TDNN for DID, and (iii) Extensive experimentation to show the effectiveness of the learnt representations for DID in low-resource conditions.

The rest of the paper is structured as follows. Section II describes the related works. Section III describes our proposed approach for learning DID-specific representation. Section IV provides a description of the datasets used and the results of the experimental study, followed by the conclusions in section V.

II. RELATED WORK

In the recent times, there is an upsurge in the use of voice-interactive systems. The foundation for all these models is LID. However, dialectical variations are known to reduce the efficacy of these systems [2], [21], [22], [23]. Therefore, there is an increasing interest in identifying the dialects of the language. However, so far very less progress has been made for the dialects of Indian languages mainly because of scarcity of data. The pre-trained feature extractors such as bottleneck features (BNF) [10], wav2vec 2.0 [12], etc., have been used to obtain representations capturing language-specific [24], [25] and dialect-specific information [7]. Generally the features are extracted from the last layer of the pre-trained feature extractor. Then these representations are used to train various SOTA models such as x-vector, BLSTM based u-vector, Transformer based u-vector, etc. to train LID [24], [26] and DID systems [7].

Since, the wav2vec 2.0 is based on the Transformer architecture which is known to capture larger context, it is known to generate powerful representations [12]. Therefore, we explore various model configurations of wav2vec 2.0. Due to these capabilities of the wav2vec 2.0, it has been used in various speech related tasks [18], [19], [27], [28]. Recently, layer-wise characteristics captured by various layers of the wav2vec 2.0 have also been explored [17], [27], [29], [30]. It has been observed that speech characteristics are not evenly distributed across all the hidden layers, instead they follow an acoustic-linguistic hierarchy [17]. Therefore, the representations learnt from different layers are suitable for different speech related tasks [30], [31], [32].

Given the diversity of information captured by various layers of the wav2vec 2.0 pre-trained model, in the recent past, across various domains related to speech there is an attempt to aggregate the features obtained from various layers using various pooling techniques [18], [19], [33], [34]. In the domain of voice-conversion (VC), wav2vec-VC has been proposed that uses the representations from all the layers of the wav2vec 2.0 model for any-to-any VC [18] and is based on disentanglement-based approach. Similarly, a multi-fusion attentive classifier has been used for audio deepfake detection using WavLM, which is a self-supervised pre-trained model similar to wav2vec 2.0 and got encouraging results [19]. However, to the best of our knowledge nothing of this sort has been done for DID so far.

TDNN is known to capture temporal dynamics for short-utterances. However, it does not works well for capturing long-range context especially for longer utterances and performs poorly when subjected to noise, therefore various methods such as extending the depth of TDNNs by introducing skip connections [35], inserting dense layers between each pair of hidden layers [36], using filter with varying temporal resolution for more powerful context representation [37], thin TDNNs (lesser width and more depth) with split-transform-merge structure that helps the deep TDNN with limited width [38] have been employed to increase global context. These methods, however, do not consider the fusion of local and global features and thus their representation capability is limited. Recently, in [20] a novel and computationally less expensive DS-TDNN has been used in the domain of speaker verification to learn robust speaker embeddings. This model is known to learn robust speaker embeddings through 2 different branches in a parallel pattern. The global branch uses Global-Aware filter (GAF) based on FFT/IFFT along with dynamic filtering and sparse regularization and the local branch uses Res2Conv module. Besides the final fusion of the two branches, normalized element-wise summation is also employed to gradually fuse local and global features within the branch.

III. PROPOSED APPROACH

A. AGGREGATION OF wav2vec 2.0 FEATURES

Fig.1 shows the proposed architecture which essentially comprises of three modules: wav2vec 2.0 feature extractor, pooling module and the classifier. The input to this architecture is a raw audio which passes through each layer of the Transformer encoder of the pre-trained wav2vec 2.0 model, then to the pooling layer followed by the classifier. In this section, we provide a detailed description of these modules.

1) PRE-TRAINED wav2vec 2.0 FEATURE EXTRACTOR

It has a multi-layer convolutional feature encoder followed by a Transformer network. The multi-layer convolutional feature encoder takes raw audio as input and outputs a sequence of latent speech representations for T time-steps (T will remain same across layers but will be different for speech samples of different length). These latent representations are then fed to a Transformer encoder to build representations capturing information from the entire sequence [12]. The output of l-th Transformer layer is $\mathbf{H}_l = (\mathbf{h}_{1l}, \mathbf{h}_{2l}, \ldots, \mathbf{h}_{Tl})$, $\mathbf{h}_{tl} \in \mathbb{R}^{\mathbb{D}}$. The output of each of the layer of the Transformer encoder goes to the second logical unit of the proposed architecture i.e. the pooling module.

2) POOLING MODULE

In the pooling module, across the time steps the pooling is applied to get a compact representation for the particular layer, as shown in left part of the pooling module in Fig. 1. Since each layer of the Transformer encoder is known to capture different linguistic properties, to get an efficient representation we aggregate the features from all the layers $l = 1, 2, \ldots L$ of the Transformer encoder. Here, L is the number of layers in Transformer encoder. Thus, the pooling is applied across both time as well as layers to get an utterance-level embedding. Note that, for the utterances of different length, the number of time-steps T is bound to be different. Hence, to convert the representations into a fixed-dimensional representation, pooling across time is applied first. Followed by this, we aggregate the features from all layers of the Transformer encoder by applying another pooling (pooling across layers), which leads to a fixed-dimensional utterance-level representation. We propose to use two types of pooling, statistical pooling and attentive statistical pooling, which are described below.

a: STATISTICAL POOLING

Let $\mathbf{H}_l = (\mathbf{h}_{1l}, \mathbf{h}_{2l}, \ldots, \mathbf{h}_{tl}, \ldots, \mathbf{h}_{Tl})$, be the sequence of feature vectors obtained from the Transformer encoder layer l, l varying from 1, $\ldots L$. The statistics pooling layer first computes the mean ($\boldsymbol{\mu}_l$) and standard deviation ($\boldsymbol{\sigma}_l$) of all hidden layer outputs as follows.

$$\boldsymbol{\mu}_l = \frac{1}{T}\sum_{t=1}^{T} \mathbf{h}_{tl} \tag{1}$$

$$\boldsymbol{\sigma}_l = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\mathbf{h}_{tl} - \boldsymbol{\mu}_l)^2} \tag{2}$$

The mean vector and standard deviation are then concatenated together as: $\mathbf{z}_l = [\boldsymbol{\mu}_l^\top, \boldsymbol{\sigma}_l^\top]^\top$. This gives a 2x$D$

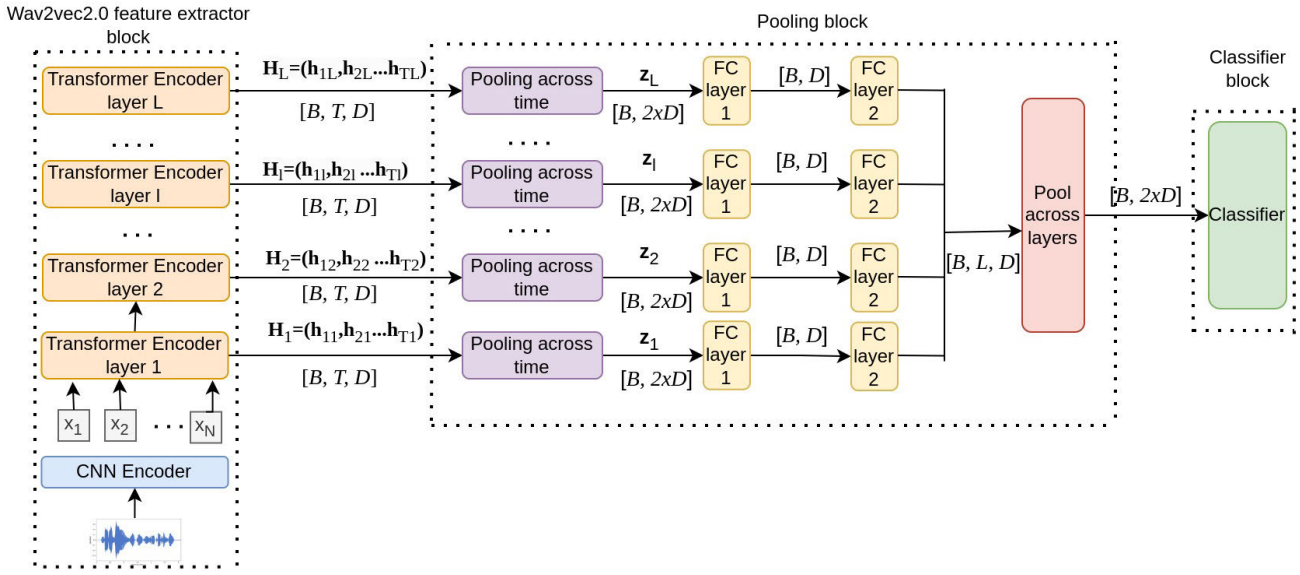


FIGURE 1. Three blocks constituting our proposed model for dialect identification. Here, *T* represents the time-steps obtained as an output of transformer encoder layer. *D* represents the feature vector dimension obtained from wav2vec 2.0 ($D = 1024$ for wav2vec 2.0 large and $D = 768$ for wav2vec 2.0 base). Since SP and ASP involve concatenation of mean and standard deviation of the feature vectors thus, the output after the pooling is 2x*D* (2x$D = 2048$ for wav2vec 2.0 large and 2x$D = 1536$ in case of wav2vec 2.0 base.) *B* denotes the batch size (batch-size used = 1). *L* represents the number of layers in the transformer encoder.

(where D is output feature dimension of wav2vec 2.0 model) dimensional vectors for each layer of the model. These representations are then given to two dense layers, as shown in Fig 1. First hidden dense layer contains D nodes and second layer contains D nodes. Output of this second layer forms a compact D-dimensional representation for layer l. The output from these fully-connected layers is stacked together to get a $[B, L, D]$ dimensional representation, where B is the batch size.

Following this, we apply pooling across the layers, where we take the mean and standard deviation of all layer-wise vectors as follows.

$$\boldsymbol{\mu} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{z}_l \tag{3}$$

$$\boldsymbol{\sigma} = \sqrt{\frac{1}{L}\sum_{l=1}^{L}(\mathbf{z}_l - \boldsymbol{\mu})^2} \tag{4}$$

When statistical pooling is applied across layers, the output is $\mathbf{u} = [\boldsymbol{\mu}^\top, \boldsymbol{\sigma}^\top]^\top$ where, $\mathbf{u}$ is a 2xD dimensional utterance-level embedding of the input speech sample, suitable for DID. This embedding is finally given to the classifier block to perform DID as shown in Fig.1.

b: ATTENTIVE STATISTICAL POOLING

In real-world scenarios, some part of the speech sample may contain significantly more dialect-discriminative contents than other parts. Hence, simple statistics pooling, where equal weight is assigned to each representation, is a suboptimal approach. Attentive statistical pooling (ASP) is a pooling method that combines the advantages of both attention and statistic pooling [39]. Thus, this approach uses higher-order statistics (standard deviation as utterance-level features) and attention mechanism. This method produces both means and standard deviations by using attention mechanism to obtain the importance of each frame. Since this approach uses higher order statistics along with attention mechanism, these are expected to capture dialect-specific information in a more efficient manner.

Let $\mathbf{H}_l = (\mathbf{h}_{1l}, \mathbf{h}_{2l}, \ldots, \mathbf{h}_{tl}, \ldots, \mathbf{h}_{Tl})$, be the sequence of feature vectors obtained from a Transformer encoder layer l, l varying from $1 \ldots L$. ASP calculates the weighted mean $\widetilde{\boldsymbol{\mu}}_l$ and weighted standard deviation $\widetilde{\boldsymbol{\sigma}}_l$ of the sequence as follows:

$$\widetilde{\boldsymbol{\mu}}_l = \sum_{tl=1}^{T} \alpha_{tl}\mathbf{h}_{tl} \tag{5}$$

$$\widetilde{\boldsymbol{\sigma}}_l = \sqrt{\sum_{t=1}^{T} \alpha_{tl}\mathbf{h}_{tl} \odot \mathbf{h}_{tl} - \widetilde{\boldsymbol{\mu}}_l \odot \widetilde{\boldsymbol{\mu}}_l} \tag{6}$$

where α_{tl} is the attention weight of the t-th frame, and $\odot$ denotes the element-wise product. Attention weights (α_{tl}) are normalized scalar values computed for each frame after passing through two linear layers, the first layer with N_a nodes and the second layer with a single node as:

$$e_{tl} = \mathbf{v}^\top f(\mathbf{W}_a^\top \mathbf{h}_{tl} + \mathbf{b}_a) + k \tag{7}$$

$$\alpha_{tl} = \frac{\exp(e_{tl})}{\sum_{\tau_l=1}^{T}\exp(e_{\tau_l})} \tag{8}$$

where e_{tl} represents scalar score for each frame-level feature, $\mathbf{W_a} \in \mathbb{R}^{D\times N_a}$ and $\mathbf{b}_a \in \mathbb{R}^{N_a}$, respectively represents the

weights and biases for a layer with N_a nodes, $\mathbf{v} \in \mathbb{R}^{N_a \times 1}$ represents the weights for the layer with a single node and k represents the bias for the layer with a single node.

A fixed length utterance-level representation ($\mathbf{z}_l$) for each of the layer is computed by concatenating $\widetilde{\boldsymbol{\mu}}_l$ and $\widetilde{\boldsymbol{\sigma}}_l$. Since each layer of the Transformer encoder is known to capture different linguistic properties, to get discriminative representations we aggregate the features from all the layers $l = 1, 2, \ldots L$ of the Transformer encoders across time-steps T using pooling as shown in left part of pooling block in Fig 1. For the utterances of different length the number of time-steps T is bound to be different, so to convert the representations into a fixed dimensional representation, pooling is applied. Thus, first for output of each layer l, l varying from 1, 2 $\ldots$. L of the Transformer encoder the pooling is applied. Then these are given as an input to 2 fully connected neural networks to convert the 2xD dimensional feature representation into D dimensional representation. This representation captures utterance-level information captured by each of the Transformer encoder layer. The representations from all the L layers are stacked together to give [B,L,D] dimensional representation. Now, pooling is applied across layers i.e. on this stacked representation by concatenating the weighted mean ($\widetilde{\boldsymbol{\mu}}$) and weighted standard deviation ($\widetilde{\boldsymbol{\sigma}}$) as follows:

$$\widetilde{\boldsymbol{\mu}} = \sum_{l=1}^{L} \alpha_l \mathbf{z}_l \tag{9}$$

$$\widetilde{\boldsymbol{\sigma}} = \sqrt{\sum_{l=1}^{L} \alpha_l \mathbf{z}_l \odot \mathbf{z}_l - \widetilde{\boldsymbol{\mu}} \odot \widetilde{\boldsymbol{\mu}}} \tag{10}$$

α_l represents the attention weight of the l-th layer. It represents the normalized score value computed for each layer after passing through two linear layers, the first layer has N_e nodes and the second layer has a single node, which is computed as follows:

$$e_l = \mathbf{v}_1^\top f(\mathbf{W}_1^\top \mathbf{z}_l + \mathbf{b}_1) + k_1 \tag{11}$$

$$\alpha_l = \frac{\exp(e_l)}{\sum_{l=1}^{L} \exp(e_l)} \tag{12}$$

where, e_l represents the scalar score for each layer, $\mathbf{W}_1 \in \mathbb{R}^{D \times N_e}$, $\mathbf{b}_1 \in \mathbb{R}^{N_e}$ represents the weights and biases for the layer with N_e nodes, $\mathbf{v}_1 \in \mathbb{R}^{N_e \times 1}$ represents the weights for the layer with a single node and k_1 represents the bias for the layer with a single node. Using ASP across the layers helps to determine the importance of each layer in obtaining dialect-specific representation. The final utterance-level embedding is obtained by concatenating $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\boldsymbol{\sigma}}$. Thus, this representation has a dimension of 2xD and is to be used for our DID system and is obtained as shown in Fig.1. This representation is fed to the classifier to obtain the final prediction.

3) CLASSIFIER BLOCK

The classifier block consists of 3 fully connected layers. The input to the classifier block is the output representation obtained by applying pooling across the layers. The first fully connected layer takes 2xD dimensional input and converts into D dimensional representation as this layer has D nodes. The second fully connected neural network also has D nodes, followed by another fully connected neural network having nodes equal to the number of dialects for DID.

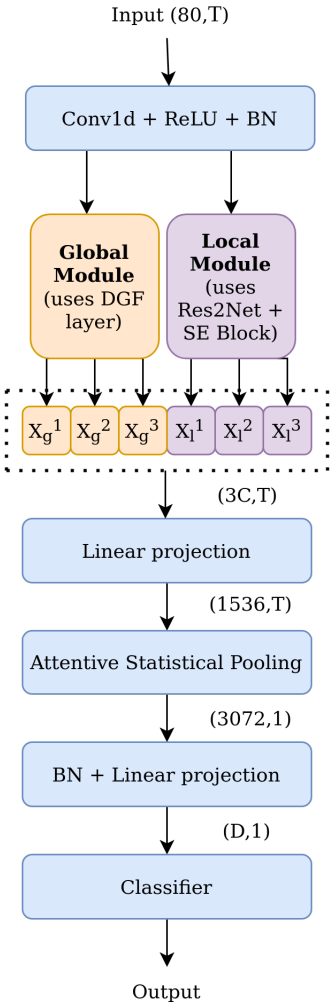


FIGURE 2. Architecture of novel DS-TDNN.

B. USING DS-TDNN TO LEARN ROBUST FEATURE REPRESENTATION

Recently, in [20] authors introduced a novel architecture called as DS-TDNN. This architecture, as shown in Fig.2 computes the global and local features in a parallel pattern. The local branch uses Res2Conv structure along with SE module [40]. The global branch introduces the GAF layer that is based on a discrete Fourier transform, a set of differentiable transform-domain filters, and the inverse discrete Fourier transform [20]. The tokens in TDNN $\mathbf{X} \in \mathbb{R}^{C \times T}$ are considered a series of discrete sequences $\mathbf{x}_i$ stacked along the channel dimension. 1D FFT is applied on individual channels,

to obtain the corresponding spectrum $\mathbf{X}_f$ as.

$$\mathbf{X}_f = \text{Concat}\big(\mathcal{F}[\mathbf{x}_1], \mathcal{F}[\mathbf{x}_2], \ldots, \mathcal{F}[\mathbf{x}_C]\big) \in \mathbb{C}^{C\times T} \quad (13)$$

, where $\mathcal{F}$ denotes the discrete fourier transform. Since $\mathbf{X}$ is a real tensor it is conjugate symmetric, only half of the $\mathbf{X}_f$ is needed for the real-time processing. The Fourier transform condenses all channel information into the spectrum, allowing efficient modeling of channel-wise global context. This is achieved using an element-wise multiplication between the spectrum and a differentiable filter $\mathbf{F} \in \mathbb{C}^{C\times\lfloor T/2\rfloor}$.

$$\tilde{\mathbf{X}}_r = \mathbf{F} \odot \mathbf{X}_r \quad (14)$$

, where $\mathbf{X}_r$ is a complex tensor using only half of $\mathbf{X}_f$, and $\odot$ represents the Hadamard product. The inverse FFT is applied to the modulated spectrum $\tilde{\mathbf{X}}_r$ to transform it back to the time domain, updating the tokens as:

$$\mathbf{X} \leftarrow \mathcal{F}_r^{-1}[\tilde{\mathbf{X}}_r] \quad (15)$$

To further enhance the performance of GAF layer, dynamic filtering strategy is used that helps the GAF layer to dynamically adapt to the input. Basically, independently K GAF layers are applied and are combined using element-wise summation. The dynamic scores $\mathbf{w}$, computed as:

$$\mathbf{w} = \underbrace{\text{Softmax}(\text{FC}_2(\text{ReLU}(\text{FC}_1(\text{GAP}(\mathbf{X})))))}_{\text{Attention FN}} \in \mathbb{R}^{1\times K} \quad (16)$$

, where FC_1, FC_2 represent fully connected layers and GAP represent the global average pooling. The dynamic scores $\mathbf{w}$ are assigned to each of these K filters. The dynamic global filter (DGF) represents normalized linear combination of filters. DGF is represented as

$$\mathbf{F}_d = w_1\mathbf{F}_1 + w_2\mathbf{F}_2 + \cdots + w_K\mathbf{F}_K \in \mathbb{C}^{C\times\lceil T/2\rceil} \quad (17)$$

Finally, the modulated spectrum $\tilde{\mathbf{X}}_r$ is obtained by element-product with DGF ($\mathbf{F}_d$).

$$\tilde{\mathbf{X}}_r = \mathbf{F}_d \odot \mathbf{X}_r \quad (18)$$

Further sparse regularisation is applied to prevent overfitting due to increased parameters. Sparse regularisation is applied by an element-wise multiplication with a random sparse-channel mask $\mathbf{M} \in \mathbb{R}^{C\times\lfloor T/2\rfloor}$, i.e.,

$$\mathbf{F}_s = \mathbf{M} \odot \mathbf{F}_d \quad (19)$$

The mask is applied directly to the DGF. $\mathbf{F}_s$ represents dynamic global-aware filter with sparse regularization. The information is captured using 2 separate branches: local and global. The features from the local and global branch are fused together not only at the end of the branches but also within each branch. These capture complimentary information and have different-scale receptive field.

IV. DATASET

We obtain the dataset for the dialects of Konkani, Marathi, Kannada and Tamil from Linguistic Data Consortium for Indian languages (LDC-IL) Konkani raw speech corpus [41], Marathi raw speech corpus [41], Kannada raw speech corpus [41], [42], Tamil raw speech corpus [41], [43]. The data is available in various domains. We consider "sentence" and "Contemporary Text (News)" level utterances from the available dataset. The details of the Konkani, Marathi, Kannada and Tamil dataset can be seen in Table 1, Table 2, Table 3 and Table 4, respectively.[1] The dataset is fairly balanced with speakers from varied age groups, genders, etc. The dataset used in our work has a wide range of age groups i.e. 16 to 20, 21 to 50 and above 51. Furthermore, the dataset is also balanced in terms of the gender of the speakers. Due to all these, we believe that the models built are free from any significant bias. Thus, the model is applicable across diverse speaker profiles. All the speech samples used in this work have a duration between 3 to 10 seconds. The Contemporary Text which originally had longer speech samples has been split into a duration of 3 to 10 seconds. The train-test split considered in this work is in the ratio of 7:3.

TABLE 1. Details of Konkani dialects dataset.

Age group	No. of speech utterances	Dialect (region) wise distribution		
		Karwari	North Goa	South Goa
		Male / Female	Male / Female	Male / Female
16 To 20	5190	356 / 250	869 / 1247	1319 / 1140
21 To 50	23079	3466 / 3188	2962 / 5336	4489 / 3638
Above 51	9744	2239 / 1869	645 / 839	1970 / 2182
Total	**38013**	**6061 / 5307**	**4476 / 7422**	**7778 / 6960**

TABLE 2. Details of Marathi dialects dataset.

Age group	No. of speech utterances	Dialect (region) wise distribution	
		Marathwada	Puneri
		Male / Female	Male / Female
16 To 20	1349	347 / 364	310 / 328
21 To 50	7552	1800 / 1871	1789 / 2092
Above 51	3454	948 / 949	837 / 720
Total	**12355**	**3095 / 3184**	**2936 / 3140**

V. EXPERIMENTAL STUDY

In this section, we evaluate the performance of the proposed approaches and compare them with the baseline. The performances of all DID systems are measured in terms of accuracy and equal error rate (EER).

A. BASELINE SYSTEMS

In this study, we use two baseline systems to evaluate the effectiveness of the proposed approaches. First one is the state-of-the-art x-vector network [44]. It contains a

[1]The dataset can be accessed from the website of Linguistic Data Consortium of India (LDCIL) at https://data.ldcil.org/speech/speech-raw-corpus

TABLE 3. Details of Kannada dialects dataset.

Age group	No. of speech utterances	Dialect (region) wise distribution			
		Hyderabad Karnataka	Canara	Mumbai Karnataka	Old Mysore
		Male / Female	Male / Female	Male / Female	Male / Female
16 to 20	5640	655 / 774	660 / 668	716 / 696	686 / 775
21 to 50	28154	3267 / 3385	3383 / 3307	3291 / 3507	3957 / 4057
Above 51	13282	1527 / 1651	1505 / 1580	1594 / 1598	1922 / 1905
Total	**47076**	**5449 / 5810**	**5548 / 5555**	**5601 / 5801**	**6565 / 6737**

TABLE 4. Details of tamil dialects dataset.

Age group	No. of speech utterances	Dialect (region) wise distribution					
		Madurai Tamil	Nellai Tamil	Kongu Tamil	Salem Tamil	Thanjai Tamil	Kumari Tamil
		Male / Female	Male / Female	Male / Female	Male / Female	Male / Female	Male / Female
16 to 20	4208	0 / 173	384 / 238	129 / 378	143 / 0	676 / 1336	538 / 213
21 to 50	30824	1704 / 2144	991 / 940	2697 / 3491	1143 / 248	7818 / 7730	951 / 967
Above 51	8391	530 / 923	1096 / 648	450 / 707	1893 / 1267	228 / 177	223 / 249
Total	**43423**	**2234 / 3240**	**2471 / 1826**	**3276 / 4576**	**3179 / 1515**	**8722 / 9243**	**1712 / 1429**

TABLE 5. Results obtained on Konkani dataset.

Front-end	Features used	Accuracy	EER
x-vector based system	MFCC	68.58%	28.65%
w2v-single system	wav2vec 2.0 large (3rd)	82.6%	14.89%
	wav2vec 2.0 base (5th)	76.13%	21.18%
w2v-uniform system	wav2vec 2.0 large	89.19%	6.15%
	wav2vec 2.0 base	88.65%	7.12%
w2v-attn system	wav2vec 2.0 large	**92.53%**	**4.21%**
	wav2vec 2.0 base	91.56%	4.98%
DS-TDNN system	Mel-spectrogram	89.95%	5.97%

TABLE 6. Results obtained on Marathi dataset.

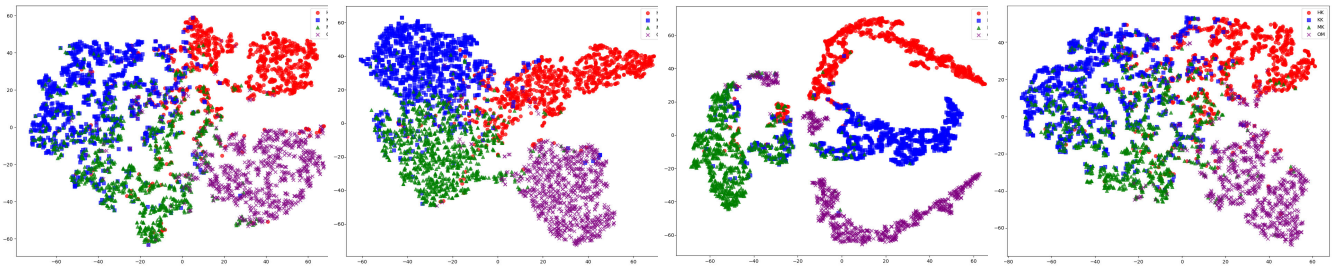
Front-end	Features used	Accuracy	EER
x-vector based system	MFCC	78.01%	17.78%
w2v-single system	wav2vec 2.0 large	87.25%	7.18%
	wav2vec 2.0 base	85.41%	9.41%
w2v-uniform system	wav2vec 2.0 large	93.45%	2.94%
	wav2vec 2.0 base	89.65%	4.01%
w2v-attn system	wav2vec 2.0 large	**97.51%**	**1.07%**
	wav2vec 2.0 base	93.75%	2.88%
DS-TDNN system	Mel-spectrogram	96.25%	1.54%

TDNN based front-end to process the input frame-level features. Following this, it uses statistics pooling to obtain an utterance-level vector of the input speech (denoted as x-vector). Lastly, it uses a classifier to predict the dialect in the speech. The whole network (which contains the TDNN-based front-end, pooling layer and the classifier), is trained in an end-to-end manner using cross-entropy loss function. For this system, we use the 13-dimensional MFCC features along with delta and delta-delta coefficients as the input (39-dimensional). The results obtained for this system (denoted as *x-vector*) for the dialects of Konkani, Marathi, Kannada and Tamil is given in 1^{st} row of Table 5, Table 6, Table 7 and Table 8 respectively.

FIGURE 3. Architecture for *w2v-single*.

Another baseline system is the simplified version of DID system shown in Figure 1, which uses features from only a single layer of wav2vec 2.0 network as the input representation. Such single-layer based representation system allows us to gauge the effectiveness of proposed approaches which use features from all layers of wav2vec 2.0. Specifically, as shown in Figure 3, for a given speech sample, we take features from a pre-defined layer (3^{rd} layer for wav2vec 2.0 large and 5^{th} layer for wav2vec 2.0 base). This selection of layer is done based on the previous study [17]. Note that, for a given speech sample, the representations obtained from any of the layers of wav2vec 2.0 has T time-steps. Since T depends on the length of the input speech sample, we need to compute a fixed-dimensional representation before feeding it to the classifier. In our case, we apply an attention-based statistical pooling (which considers the relevance of features in individual time-steps in determining dialect) to obtain a fixed-dimensional representation. The obtained utterance-level vector is then classified using the DID classifier. The classifier contains a set of two dense layers (with 1024 nodes in each), followed by the output layer with *softmax* activation. We denote this architecture as *w2v-single*.

Results obtained for this system is given in 2^{nd} row of Table 5 (for Konkani), Table 6 (for Marathi), Table 7 (for Kannada) and Table 8 (for Tamil). From the results, it can be seen that, both *x-vector* and *w2v-single* systems have performed reasonably well on both datasets. Among the two, the *w2v-single* system has performed better compared to the

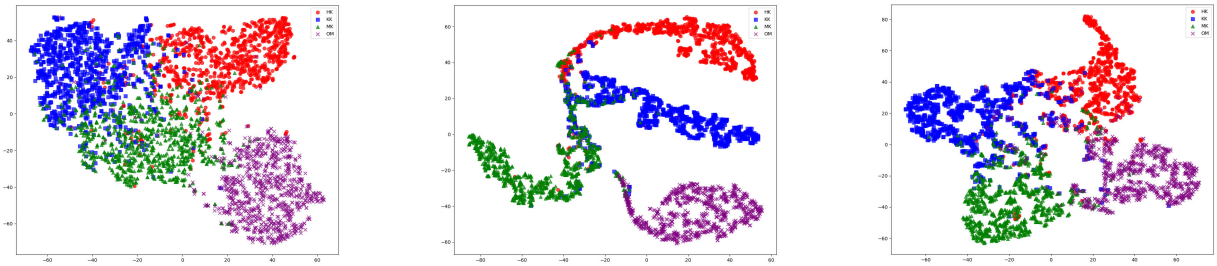


(a) t-SNE plot of representation from *w2v-single* (wav2vec 2.0 Large)

(b) t-SNE plot of representation from *w2v-uniform* (wav2vec 2.0 Large)

(c) t-SNE plot of representation from *w2v-attn* (wav2vec 2.0 Large)

(d) t-SNE plot of representation from *w2v-single* (wav2vec 2.0 Base)



(e) t-SNE plot of representation from *w2v-uniform* (wav2vec 2.0 Base)

(f) t-SNE plot of representation from *w2v-attn* (wav2vec 2.0 Base)

(g) t-SNE plot of representation from DS-TDNN

FIGURE 4. t-SNE plots for dialects of Kannada (embeddings obtained from the layer before the classification layer).

TABLE 7. Results obtained on Kannada dataset.

System	Features used	Accuracy	EER
x-vector based system	MFCC	70.12%	21.96%
w2v-single system	wav2vec 2.0 large (3rd)	83.14%	11.26%
	wav2vec 2.0 base (5th)	79.28%	13.97%
w2v-uniform system	wav2vec 2.0 large	90.67%	6.19%
	wav2vec 2.0 base	86.6%	8.56%
w2v-attn system	wav2vec 2.0 large	**92.45%**	**4.99%**
	wav2vec 2.0 base	91.65%	5.51%
DS-TDNN system	Mel-spectrogram	89.95%	6.68%

former one. This is mainly because, the *x-vector* uses simple MFCC features, whereas, the *w2v-single* system uses features obtained from wav2vec2.0 model.

Next, we discuss the experiments on the proposed approaches.

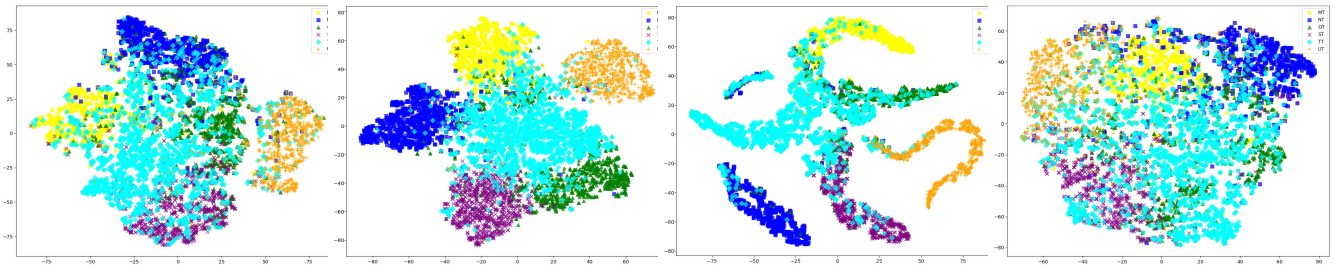
B. STATISTICAL POOLING-BASED AGGREGATION

In this case, we consider the features obtained from all layers of the wav2vec 2.0 and combine them using statistical pooling. Specifically, we obtain features from all L layers of the Transformer encoder of wav2vec model, where $L = 12$ for base model and $L = 24$ for the large model. The output of each of the L layers has $[B, T, D]$ dimension, where B is the batch size ($B = 1$), T is the number of time-steps and D is the dimension of feature vector ($D = 1024$ for large model and $D = 768$ for base model). Note that, for the representations obtained from each of the layers, pooling is first applied across time to obtain a fixed-dimensional representation. Then again statistical pooling is applied, this time across layers, as shown in Figure 1.

For each layer, we get a vector of 2xD dimensions (obtained by concatenating D-dimensional mean and D-dimensional standard-variance, after pooling across time). At the output of each of the pooling layer there are 2 fully connected layers that convert the $[B, 2 \times D]$ representation into $[B, D]$ dimensional representation. In this way we obtain representations from all the L layers. All L layer representations are then stacked, and again statistical pooling is applied, across L layers. This pooling results in a $[B, 2\times D]$ dimensional vector, which are then processed by 2 fully connected layers, each having D nodes. The output of the second fully connected layer (D-dimensional vector) is then processed by the output layer, having number of nodes equal to number of dialects.

Results obtained for this system (denoted as *w2v-uniform*) is given in 3^{rd} row of the Table 5 for the dialects of Konkani, Table 6 for the dialects of Marathi, Table 7 for the dialects of Kannada and Table 8 for the dialects of Tamil, respectively. Results obtained show that using aggregation of features from various layers of wav2vec 2.0 has shown significant improvement in performance compared to both baselines.

The t-SNE plots (of the embeddings obtained from the layer before the classification layer) in Figure 4 give some insights on the effectiveness of *w2v-uniform*. In this, Figure 4(b) represents the utterance-level embeddings obtained from *w2v-uniform*, which are obtained after the statistics pooling across layers. Here, we have used test samples from Kannada language, where different colours indicate different

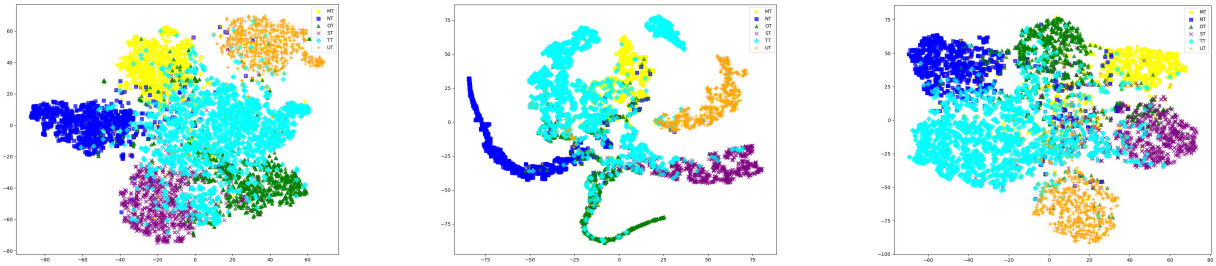


(a) t-SNE plot of embeddings from *w2v-single* (wav2vec 2.0 Large)

(b) t-SNE plot of embeddings from *w2v-uniform* (wav2vec 2.0 Large)

(c) t-SNE plot of embeddings from *w2v-attn* (wav2vec 2.0 Large)

(d) t-SNE plot of embeddings from *w2v-single* (wav2vec 2.0 Base)

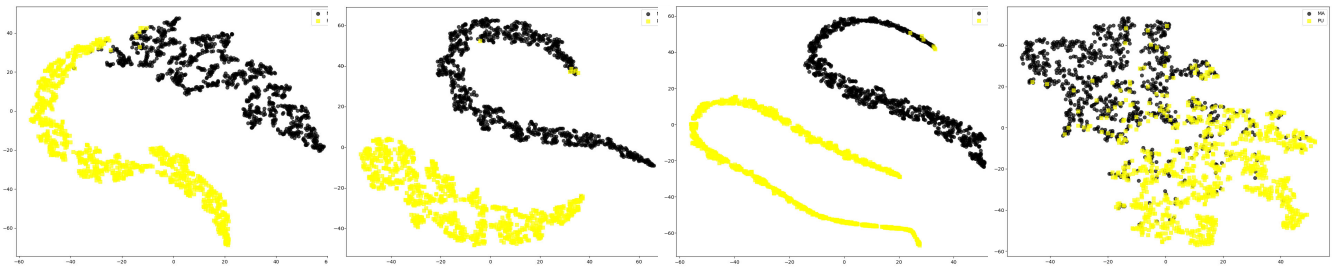


(e) t-SNE plot of embeddings from *w2v-uniform* (wav2vec 2.0 Base)

(f) t-SNE plot of embeddings from *w2v-attn* (wav2vec 2.0 Base)

(g) t-SNE plot of representation from DS-TDNN

FIGURE 5. t-SNE plots for dialects of tamil (embeddings obtained from the layer before the classification layer).

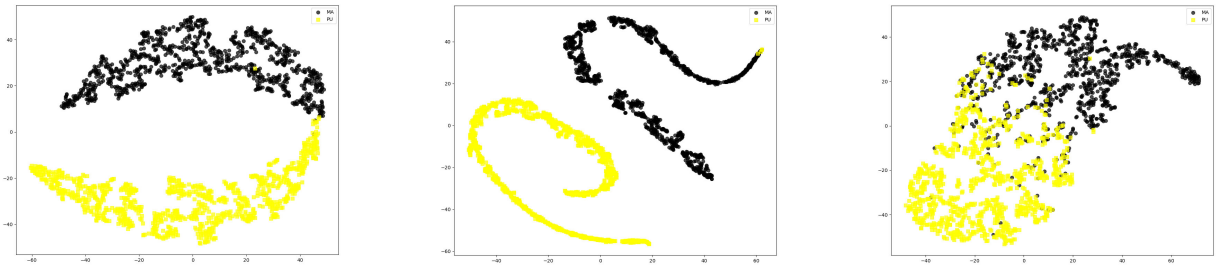


(a) t-SNE plot of embeddings from *w2v-single* (wav2vec 2.0 Large)

(b) t-SNE plot of embeddings from *w2v-uniform* (wav2vec 2.0 Large)

(c) t-SNE plot of embeddings from *w2v-attn* (wav2vec 2.0 Large)

(d) t-SNE plot of embeddings from *w2v-single* (wav2vec 2.0 Base)



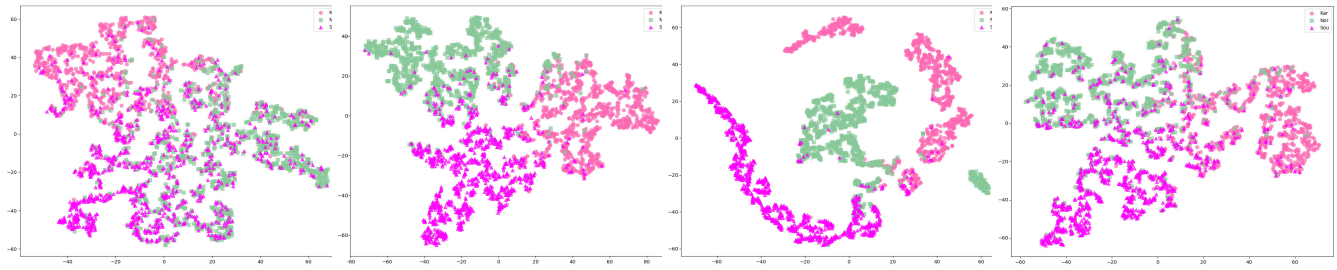
(e) t-SNE plot of embeddings from *w2v-uniform* (wav2vec 2.0 Base)

(f) t-SNE plot of embeddings from *w2v-attn* (wav2vec 2.0 Base)

(g) t-SNE plot of representation from DS-TDNN

FIGURE 6. t-SNE plots for dialects of Marathi (embeddings obtained from the layer before the classification layer).

languages. For a comparison, t-SNE plot of embeddings from *w2v-single* are also given in Figure 4(a). From the plots, it is evident that the clusters from *w2v-uniform* are more compact and separable, as it uses features from all layers of the wav2vec model. Furthermore, the embeddings from wav2vec 2.0 large form better clusters compared to wav2vec 2.0 base, indicating that the former is slightly more efficient in capturing dialect-specific cues in the input speech. Similar

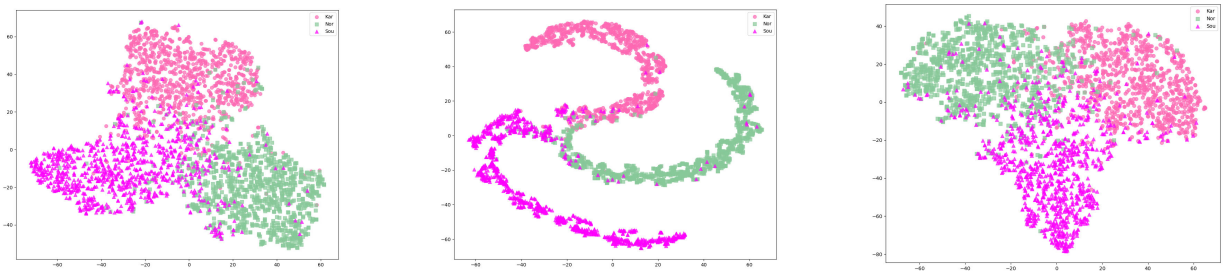


(a) t-SNE plot of representation from *w2v-single* (wav2vec 2.0 Large)

(b) t-SNE plot of representation from *w2v-uniform* (wav2vec 2.0 Large)

(c) t-SNE plot of representation from *w2v-attn* (wav2vec 2.0 Large)

(d) t-SNE plot of representation from *w2v-single* (wav2vec 2.0 Base)



(e) t-SNE plot of representation from *w2v-uniform* (wav2vec 2.0 Base)

(f) t-SNE plot of representation from *w2v-attn* (wav2vec 2.0 Base)

(g) t-SNE plot of representation from DS-TDNN

FIGURE 7. t-SNE plots for dialects of Konkani (embeddings obtained from the layer before the classification layer).

TABLE 8. Results obtained on tamil dataset.

System	Features used	Accuracy	EER
x-vector based system	MFCC	56.01%	29.78%
w2v-single system	wav2vec 2.0 large	65.25%	24.56%
	wav2vec 2.0 base	61.19%	26.60%
w2v-uniform system	wav2vec 2.0 large	76.40%	14.33%
	wav2vec 2.0 base	74.04%	17.92%
w2v-attn system	wav2vec 2.0 large	**82.33%**	**8.01%**
	wav2vec 2.0 base	76.65%	14.18%
DS-TDNN system	Mel-spectrogram	78.25%	12.31%

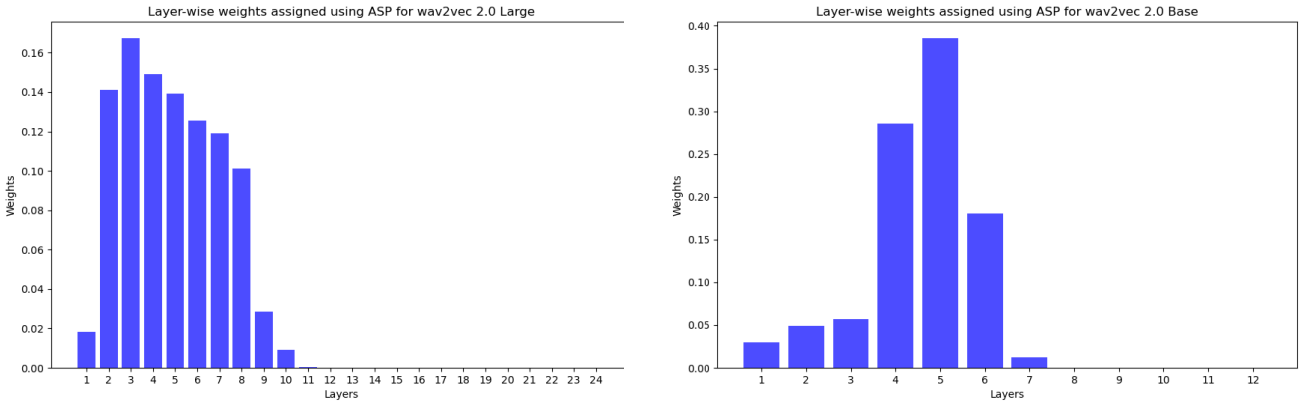
observations can be made from the t-SNE plots for the dialects of Tamil language, Marathi language and Konkani language, which are given in Figure 5, Figure 6 and Figure 7, respectively.

C. ATTENTIVE STATISTICAL POOLING-BASED AGGREGATION

In order to consider the relevance of features in individual time-steps and layers in determining the dialect-discriminative features, we use ASP based pooling strategy across both layers and time. ASP combines the benefits of both higher-order statistics and attention mechanism. The weights assigned to the layers and time-steps are in accordance with their importance in determining the dialect of the language. We denote this architecture as *w2v-attn*. The architecture of the *w2v-attn* network is similar to *w2v-uniform*. However, they differ mainly in the pooling module where in *w2v-attn* the attention-based statistical pooling replaces statistical pooling (used in *w2v-uniform*). Results obtained for this system, for Konkani, Marathi, Kannada and Tamil are respectively given in 4^{th} row of Table 5, Table 6, Table 7 and Table 8. From the results, it can be clearly seen that the performance obtained using *w2v-attn* is better than both *w2v-single* and *w2v-uniform*. This is because, attention based aggregation of features across both time and layers helps the system to better capture the DID-specific contents in the speech.

Figure 4(c) shows the t-SNE of utterance-level embeddings obtained from *w2v-attn* for dialects of Kannada language (using features obtained from wav2vec 2.0 Large model). Similarly, Figure 5(c), Figure 7(c), Figure 6(c) shows the t-SNE of embeddings from Tamil dialects, Konkani dialects and Marathi dialects (features from wav2vec 2.0 Large model). Plots obtained for wav2vec 2.0 Base model are also given in the same Figure for reference. It is seen that, in case of both Large and Base models, the clusters using *w2v-attn* are slightly more compact and clearly separable when compared to those obtained for *w2v-uniform* and *w2v-single*.

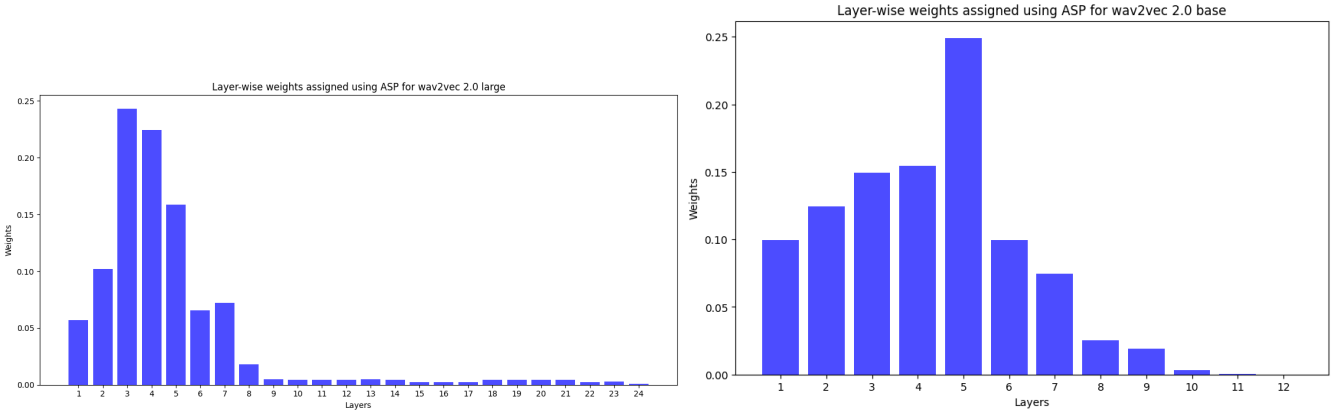
In order to see how the *w2v-attn* model assigns different weights to different layers of wav2vec2.0, we plot the attention weights distribution across the layers (averaged for all test samples). Figure 8(a) shows the average attention weights versus the layers of wav2vec 2.0 large, for dialects of Kannada. We observe that the highest weight has been assigned to the representations from the 3^{rd} layer and an almost equal weight to the representations from the 2^{nd}, 4^{th} and 5^{th} layer. A bit lesser weights are assigned to the 6^{th}, 7^{th} and 8^{th} layer. Very less weights are assigned



(a) wav2vec 2.0 Large model

(b) wav2vec 2.0 Base model

FIGURE 8. **Attention weights assigned to various layers of wav2vec 2.0 models for dialects of Kannada language.**



(a) wav2vec 2.0 Large model

(b) wav2vec 2.0 Base model

FIGURE 9. **Attention weights assigned to various layers of wav2vec 2.0 models for dialects of tamil language.**

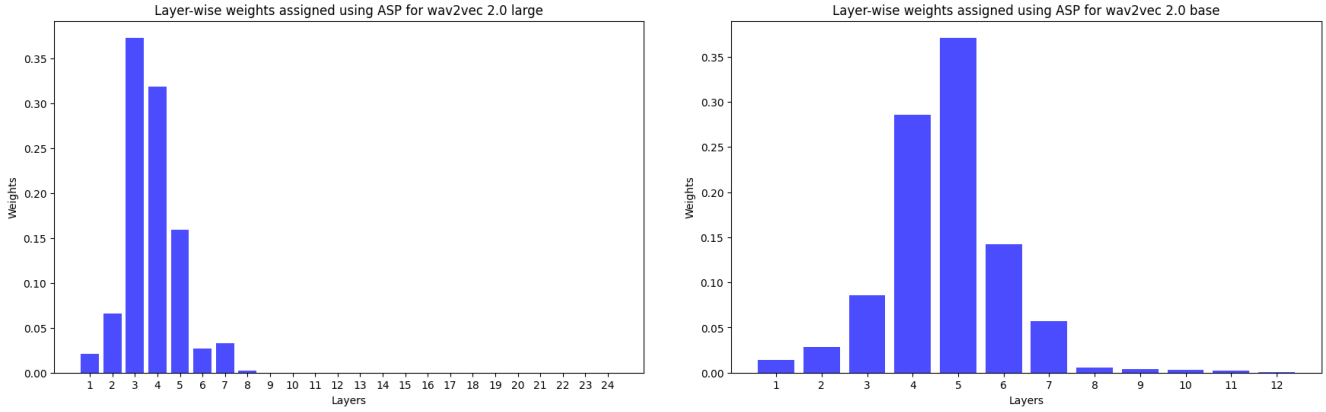
to representations from layer 9^{th} and 10^{th}, with almost negligible weights to the layers from 11^{th} to 24^{th} layer. Similarly, we plot the attention weights assigned to various layers of wav2vec 2.0 base model in Figure 8(b). We observe that the highest attention weight has been assigned to the 5^{th} layer. The 4^{th} layer is assigned a bit lesser weight followed by 6^{th} layer. Very less weights are assigned to 1^{st}, 2^{nd}, 3^{rd} and 7^{th} layer. Almost negligible weights are assigned to the 8^{th} to 12^{th} layer. From both these Figures, it is evident that, the *w2v-attn* model assigned more weights to the initial layers of the wav2vec 2.0 model.

We also plot the attention weights versus the layers for wav2vec 2.0 large for dialects of Tamil in Figure 9. For wav2vec 2.0 large model, we observe that the highest weight has been assigned to representations from the 3^{rd} layer, a little lesser weights to the representations from the 4^{th} and 5^{th} layer while the layers from the 11^{th} to the 24^{th} have been assigned negligible weights. Almost equal weights have been assigned to 1^{st}, 2^{nd} layer, while lesser weights are assigned to 6^{th}, 7^{th}, 8^{th} and 18^{th} layers. Similarly, we plot weights assigned to various layers of wav2vec 2.0 base model. We observe that the 5^{th} layer has been assigned the highest weight, followed by the 3^{rd} layer. Almost equal weights have been assigned to the 1^{st}, 2^{nd}, 3^{rd}, 4^{th}, 6^{th} and 7^{th} layers while the rest of the layers have been assigned negligible weights. Thus, similar to the dialects of Kannada, it is evident that the *w2v-attn* model assigned more weights to the initial layers of the wav2vec 2.0 model.

Similarly, we plot the attention weights assigned to different layers of wav2vec 2.0 for dialects of Marathi and Konkani in Figure 11 and Figure 10 respectively. It can be seen that the 3^{rd} layer for the wav2vec 2.0 large model and the 5^{th} layer of wav2vec 2.0 base model has been assigned the highest weight for the dialects of all the languages.

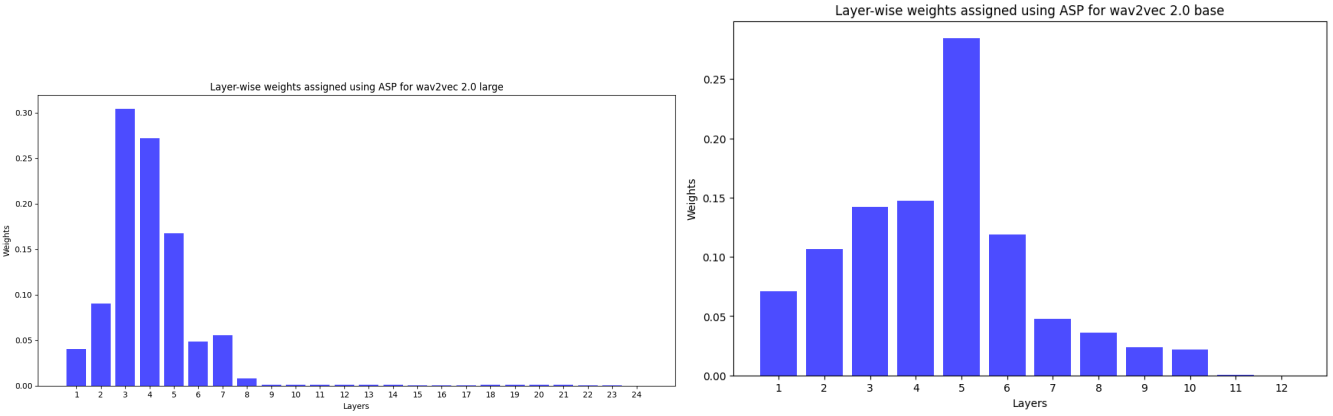
D. REPRESENTATIONS USING DS-TDNN

Here, we discuss the experiments and results for the DS-TDNN based DID sysem. DS-TDNN captures both local and global DID-specific information present in the speech sample in a parallel fashion. This architecture has two input



(a) wav2vec 2.0 Large model (b) wav2vec 2.0 Base model

FIGURE 10. Attention weights assigned to various layers of wav2vec 2.0 models for dialects of Konkani language.



(a) wav2vec 2.0 Large model (b) wav2vec 2.0 Base model

FIGURE 11. Attention weights assigned to various layers of wav2vec 2.0 models for dialects of Marathi language.

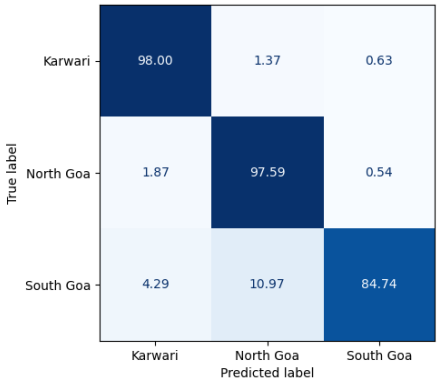


FIGURE 12. Confusion matrix for *w2v-attn* system using wav2vec 2.0 large for dialects of Konkani.

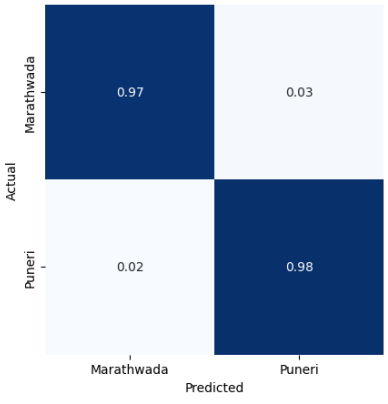


FIGURE 13. Confusion matrix for *w2v-attn* system using wav2vec 2.0 large for dialects of Marathi.

branches and a unique fusion strategy for fusing the features from these two branches. Results obtained for this system (denoted as *DS-TDNN*) are shown in the last row of Table 5, Table 6, Table 7 and Table 8, respectively for Konkani, Marathi, Kannada and Tamil languages.

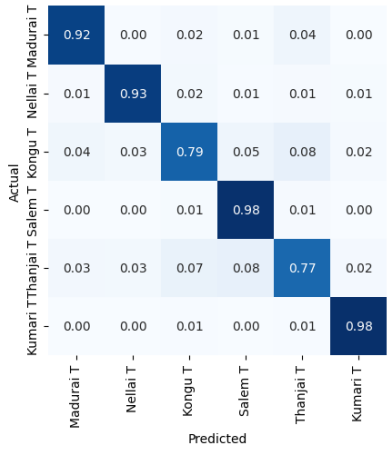


FIGURE 14. Confusion matrix for *w2v-attn* system using wav2vec 2.0 large for dialects of tamil.

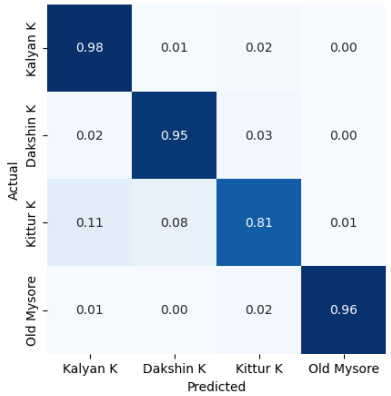


FIGURE 15. Confusion matrix for *w2v-attn* system using wav2vec 2.0 large for dialects of Kannada.

Here, we compare the results for representations obtained using DS-TDNN with the baselines and our proposed approaches (*w2v-uniform* and *w2v-attn*). It can be observed from the results in the Table 5 (for Konkani), Table 6 (for Marathi), Table 7 (for Kannada) and Table 8 (for Tamil) and the t-SNE plots in Figure 4(g) (for Kannada), Figure 5(g) (for Tamil), Figure 6(g) (for Marathi) and Figure 7(g) (for Konkani) that the representations captured using DS-TDNN perform better than both the baselines *w2v-single* and *x-vector*. This is because, unlike traditional x-vector based system which uses only one input arm to process the input features, the DS-TDNN has two input arms to capture local and global features. Moreover, we use hand-crafted MFCC features as an input into the x-vector based DID system instead of wav2vec 2.0 based powerful representations.

For the dialects of all the four languages, Konkani, Marathi, Kannada and Tamil, DS-TDNN based representations perform better than representations obtained using wav2vec 2.0 base with *w2v-uniform* and *w2v-attn* and both the baselines. Compared to results obtained for wav2vec 2.0 large, it can be clearly seen that the DS-TDNN performs better as compared to *w2v-single*. However, it performs slightly poorer than the *w2v-uniform* and poorly as compared to *w2v-attn*. To enhance the interpretability of the results we plot the confusion matrices for the DID of the four languages, Konkani, Marathi, Tamil and Kannada, in the Fig. 12, Fig. 13, Fig. 14 and Fig. 15, respectively.

VI. CONCLUSION

In this paper, we explored various methods for obtaining representations for the task of dialect identification. We use two baselines to evaluate the performance of our proposed systems. We firstly extract the MFCC features and feed them to the state-of-the-art x-vector network. This acts as the first baseline. Secondly, we obtain representations from a single layer of the pre-trained wav2vec 2.0 large and wav2vec 2.0 base model. These feature vectors are aggregated across time using attentive statistical pooling. This acts as the second baseline. We propose to obtain the representations from all the layers and across time-steps. These representations are aggregated using various pooling strategies. These include statistical pooling and attentive statistical pooling. We compare these aggregations to the baseline to show the effectiveness of the representations learnt by considering all the layers of wav2vec 2.0 instead of a single layer for obtaining the representations. We also conclude that the attention and higher-order statistics based attentive statistical pooling based aggregation is better at generating the dialect-specific information. Further, wav2vec 2.0 large is better than wav2vec 2.0 base for generating such representations. We then explore DS-TDNN for generating the representations suitable for dialect identification. It uses two branches and a unique fusion strategy to fuse features from these two branches. All the proposed approaches perform better than the baseline and perform reasonably well at capturing the dialect-specific information in low-resource condition.

REFERENCES

[1] J. K. Chambers and P. Trudgill, *Dialectology*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[2] Y. Lei and J. H. L. Hansen, "Dialect classification via text-independent training and testing for Arabic, Spanish, and Chinese," *IEEE Trans. Audio, Speech, Language Process.*, vol. 19, no. 1, pp. 85–96, Jan. 2011.

[3] D. Wang, S. Ye, X. Hu, S. Li, and X. Xu, "An end-to-end dialect identification system with transfer learning from a multilingual automatic speech recognition model," in *Proc. Interspeech*, Aug. 2021, pp. 3266–3270.

[4] N. B. Chittaragi and S. G. Koolagudi, "Automatic dialect identification system for Kannada language using single and ensemble SVM algorithms," *Lang. Resour. Eval.*, vol. 54, no. 2, pp. 553–585, Jun. 2020.

[5] A. Etman and A. A. L. Beex, "Language and dialect identification: A survey," in *Proc. SAI Intell. Syst. Conf. (IntelliSys)*, Nov. 2015, pp. 220–231.

[6] Z. Chen, S. Chen, Y. Wu, Y. Qian, C. Wang, S. Liu, Y. Qian, and M. Zeng, "Large-scale self-supervised speech representation learning for automatic speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 6147–6151.

[7] S. Monteiro, A. Angra, H. Muralikrishna, V. Thenkanidiyoor, and A. D. Dileep, "Exploring the impact of different approaches for spoken dialect identification of konkani language," in *Proc. Int. Conf. Speech Comput.*, Jan. 2023, pp. 461–474.

[8] K. R. Bagadi and C. M. R. Sivappagari, "An evolutionary optimization method for selecting features for speech emotion recognition," *TELKOMNIKA, Telecommun. Comput. Electron. Control*, vol. 21, no. 1, p. 159, Feb. 2023.

[9] M. Tzudir, S. Baghel, P. Sarmah, and S. R. M. Prasanna, "Analyzing RMFCC feature for dialect identification in ao, an under-resourced language," in *Proc. Nat. Conf. Commun. (NCC)*, May 2022, pp. 308–313.

[10] R. Fér, P. Matějka, F. Grézl, O. Plchot, K. Veselý, and J. H. Černocký, "Multilingually trained bottleneck features in spoken language recognition," *Comput. Speech Lang.*, vol. 46, pp. 252–267, Nov. 2017.

[11] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "Wav2vec: Unsupervised pre-training for speech recognition," 2019, *arXiv:1904.05862*.

[12] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2020, pp. 12449–12460.

[13] S. Miwa and A. Kai, "Dialect speech recognition modeling using corpus of Japanese dialects and self-supervised learning-based model XLSR," in *Proc. INTERSPEECH*, Aug. 2023, pp. 4928–4932.

[14] A. M. Warohma, P. Kurniasari, S. Dwijayanti, Irmawan, and B. Y. Suprapto, "Identification of regional dialects using mel frequency cepstral coefficients (MFCCs) and neural network," in *Proc. Int. Seminar Appl. Technol. Inf. Commun.*, Sep. 2018, pp. 522–527.

[15] A. Hanani and R. Naser, "Spoken Arabic dialect recognition using X-vectors," *Natural Lang. Eng.*, vol. 26, no. 6, pp. 691–700, Nov. 2020.

[16] R. Shen, Y. Li, H. Gu, Y. Wang, J. Huang, and Q. She, "Self-supervised learning representations for dialect identification with sparse transformers," in *Proc. 8th Int. Conf. Control Eng. Artif. Intell.*, Jan. 2024, pp. 1–6.

[17] A. Pasad, J.-C. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 914–921.

[18] J. Lim and K. Kim, "Wav2vec-VC: Voice conversion via hidden representations of Wav2vec 2.0," in *Proc. ICASSP - IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 10326–10330.

[19] Y. Guo, H. Huang, X. Chen, H. Zhao, and Y. Wang, "Audio deepfake detection with self-supervised wavlm and multi-fusion attentive classifier," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 12702–12706.

[20] Y. Li, J. Gan, X. Lin, Y. Qiu, H. Zhan, and H. Tian, "DS-TDNN: Dual-stream time-delay neural network with global-aware filter for speaker verification," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 32, pp. 2814–2827, 2024.

[21] V. Gupta and P. Mermelstein, "Effects of speaker accent on the performance of a speaker-independent, isolated-word recognizer," *J. Acoust. Soc. Amer.*, vol. 71, no. 6, pp. 1581–1587, Jun. 1982.

[22] M. A. Zissman, T. P. Gleason, D. M. Rekart, and B. L. Losiewicz, "Automatic dialect identification of extemporaneous conversational, Latin American Spanish speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. Conf.*, vol. 2, Jul. 1996, pp. 777–780.

[23] C. Huang, T. Chen, S. Li, E. Chang, and J. Zhou, "Analysis of speaker variability," in *Proc. 7th Eur. Conf. Speech Commun. Technol. (Eurospeech)*, Sep. 2001, pp. 1377–1380.

[24] S. Gupta, D. A. Dinesh, and P. Rajan, "Noise-robust spoken language identification using language relevance factor based embedding," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Jan. 2021, pp. 644–651.

[25] H. Muralikrishna, P. Sapra, A. Jain, and D. A. Dinesh, "Spoken language identification using bidirectional LSTM based LID sequential senones," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2019, pp. 320–326.

[26] S. Kapoor, D. A. Dinesh, and P. Rajan, "Spoken language identification in unseen target domain using within-sample similarity loss," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 7223–7227.

[27] Z. Fan, M. Li, S. Zhou, and B. Xu, "Exploring wav2vec 2.0 on speaker verification and language identification," 2020, *arXiv:2012.06185*.

[28] N. Vaessen and D. A. Van Leeuwen, "Fine-tuning Wav2Vec2 for speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 7967–7971.

[29] J. Shah, Y. Kumar Singla, C. Chen, and R. Ratn Shah, "What all do audio transformer models hear? Probing acoustic representations for language delivery and its structure," 2021, *arXiv:2101.00387*.

[30] J. Monaghan, A. Sebastian, N. Chong-White, V. Zhang, V. Easwar, and P. Kitterick, "Automatic detection of hearing loss from children's speech using wav2vec 2.0 features," in *Proc. Interspeech*, Sep. 2024, pp. 892–896.

[31] X. Chang, T. Maekaku, P. Guo, J. Shi, Y.-J. Lu, A. S. Subramanian, T. Wang, S.-W. Yang, Y. Tsao, H.-Y. Lee, and S. Watanabe, "An exploration of self-supervised pretrained representations for end-to-end speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 228–235.

[32] S.-W. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K.-T. Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H.-Y. Lee, "SUPERB: Speech processing universal performance benchmark," 2021, *arXiv:2105.01051*.

[33] S. Peng, W. Guo, H. Wu, Z. Li, and J. Zhang, "Fine-tune pre-trained models with multi-level feature fusion for speaker verification," in *Proc. Interspeech*, Sep. 2024, pp. 2110–2114.

[34] Z. Pan, T. Liu, H. B. Sailor, and Q. Wang, "Attentive merging of hidden embeddings from pre-trained speech model for anti-spoofing detection," 2024, *arXiv:2406.10283*.

[35] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, "On deep speaker embeddings for text-independent speaker recognition," 2018, *arXiv:1804.10080*.

[36] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using X-vectors," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5796–5800.

[37] X. Huang, W. Zhang, X. Xu, Y. Rui-xiang, and D. Chen, "Deeper time delay neural networks for effective acoustic modelling," *J. Phys., Conf. Ser.*, vol. 1229, no. 1, May 2019, Art. no. 012076.

[38] R. Zhang, J. Wei, W. Lu, L. Wang, M. Liu, L. Zhang, J. Jin, and J. Xu, "ARET: Aggregated residual extended time-delay neural networks for speaker verification," in *Proc. Interspeech*, Oct. 2020, pp. 946–950.

[39] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," 2018, *arXiv:1803.10963*.

[40] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[41] N. Choudhary, N. Rajesha, G. Manasa, and L. Ramamoorthy, "LDC-IL raw speech corpora: An overview," *Linguistic Resour. AI/NLP Indian Lang.*, pp. 160–174, Jan. 2019.

[42] N. Choudhary and D. G. Rao, "The LDC-IL speech corpora," in *Proc. 23rd Conf. Oriental COCOSDA Int. Committee Co-Ordination Standardisation Speech Databases Assessment Techn. (O-COCOSDA)*, Nov. 2020, pp. 28–32.

[43] N. Choudhary, "LDC-IL: The Indian repository of resources for language technology," *Lang. Resour. Eval.*, vol. 55, no. 3, pp. 855–867, Sep. 2021.

[44] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using X-vectors," in *Proc. Speaker Lang. Recognit. Workshop (Odyssey)*, Jun. 2018, pp. 105–111.

ANANYA ANGRA received the joint B.E. degree in information technology from UIET, Chandigarh, Panjab University, India, in 2022. She is currently pursuing the M.Tech. degree with the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Himachal Pradesh. She is currently working with MANAS Laboratory. She is also working as a Project Associate with Indian Institute of Technology Mandi. Her research interests include mainly applications of deep learning and machine learning in speech technology and computer vision. She is mainly working in the area of dialect identification in low-resource conditions.

H. MURALIKRISHNA received the B.E. degree from Visvesvaraya Technological University, Karnataka, India, in 2009, the M.Tech. degree in digital electronics and communication from Manipal Institute of Technology, Manipal, India, in 2013, and the Ph.D. degree from the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Himachal Pradesh, India, in 2023. He is currently working as an Assistant Professor with the Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal. His research interests include machine learning and deep learning with applications in speech technology and biomedical signal processing.

DILEEP AROOR DINESH (Member, IEEE) received the B.E. degree in computer science and engineering from Gulbarga University, Bhalki, Karnataka, India, in 2000, and the M.Tech. and Ph.D. degrees in computer science and engineering from Indian Institute of Technology (IIT) Madras, in 2006 and 2013, respectively. He joined as an Assistant Professor with the School of Computing and Electrical Engineering, IIT Mandi, Himachal Pradesh, in 2013. From 2019 to 2023, he was an Associate Professor at the School of Computing and Electrical Engineering, IIT Mandi. He is currently working as an Associate Professor with the Department of Computer Science and Engineering, Indian Institute of Technology Dharwad, Karnataka. His current research interests include applied machine learning and deep learning, speech technology, spoken language identification and diarization, computer vision, machine learning for telecom, and cloud networks.

VEENA THENKANIDIYOOR (Member, IEEE) received the B.E. degree from Manipal Institute of Technology Manipal, Mangalore University, India, in 1998, the M.S. degree from Manipal Academy of Higher Education, Manipal, in 2000, and the Ph.D. degree in computer science and engineering from Indian Institute of Technology Madras, Chennai, India, in 2014. From 2013 to 2018, she was an Assistant Professor at the Department of Computer Science and Engineering, NIT Goa, India. Since 2018, she has been working as an Associate Professor with the Department of Computer Science and Engineering. Her research interests include deep learning, kernel methods, computer vision, speech processing, content-based information retrieval, and pattern recognition.