IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 12 June 2024, accepted 29 June 2024, date of publication 3 July 2024, date of current version 19 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3422380

RESEARCH ARTICLE

Spoken Language Identification in Unseen Target Domain Using Centroid Similarity Loss With Adaptive Gradient Blending

MURALIKRISHNA H[1], SUJEET KUMAR[2], DILEEP AROOR DINESH[3], (Member, IEEE), AND VEENA THENKANIDIYOOR[4], (Member, IEEE)

[1]Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India
[2]MANAS Laboratory, Indian Institute of Technology Mandi, Suran, Himachal Pradesh 175075, India
[3]Department of Computer Science and Engineering, Indian Institute of Technology Dharwad, Dharwad, Karnataka 580011, India
[4]Department of Computer Science and Engineering, National Institute of Technology Goa, Ponda 403401, India

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

ABSTRACT In this paper, we propose a centroid similarity loss (CSL) with adaptive gradient blending (AGB) (denoted as *CSL-with-AGB*) strategy to improve the generalization of a spoken language identification (LID) system to unseen target domain conditions. Unlike most of the existing approaches, the proposed CSL-with-AGB can improve the generalization even when the training dataset lacks domain-diversity. Specifically, in this approach, the LID network first analyses the input at two different temporal resolutions using a set of two embedding extractors, which allow them to generalize better by encoding complementary contents. We then propose to use the CSL to further improve the generalization of the network by encouraging the embedding extractors to learn discriminative and domain-invariant embeddings. However, application of auxiliary loss like CSL can sometimes force the two embedding extractors of the network to learn in an unbalanced way, diminishing their ability to encode complementary contents in the input. To overcome this issue, we propose to include the AGB strategy with the CSL. With the help of two auxiliary classifiers attached to the two embedding extractors, the AGB monitors and guides them to have a balanced learning, leading to enhanced performance in unseen target domain conditions.

INDEX TERMS Spoken language identification, unseen target domain, domain-mismatch, adaptive gradient blending, centroid similarity loss.

I. INTRODUCTION

Degradation in the performance of deep learning based systems in real-world conditions is a long-standing problem. Spoken language identification (LID) systems are not exceptional to this issue. Domain-mismatch, high interclass similarities and high intraclass variations in the real-world test samples are the main reasons for such degraded performance. While domain-mismatch occurs due to mismatch in channel, background conditions, etc., between training and testing samples, interclass similarities between languages arise mainly due to overlap in their phoneme set and phonotactics [1], [2], [3]. Intraclass variations are the variations within a given language class, for which, the main sources are the dialects and accents of the language [4], [5]. In order to get a satisfactory performance in real-world conditions, the LID system should be robust to these challenges.

Note that, as the target domain conditions are unknown *a priori* in real-world conditions, commonly used supervised/unsupervised domain adaptation [6], [7], [8], [9] techniques cannot be used [10]. Instead, we need to improve the generalization of the system, which reduces its vulnerability to the challenges like domain-mismatch, interclass similarities and intraclass variations in unseen target domain conditions. In general, such improved generalization will enhance the reliability of deep learning based systems.

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Gyu Kim.

II. RELATED WORK

Several approaches have been proposed in the past to improve the generalization of the system to unseen target domain conditions. For example, the domain attentive fusion based strategy [11], adversarial multi-task learning (AMTL) [12], [13] or meta-learning based [14] strategies, etc., are some previously used approaches to improve the robustness of the system to unseen target domain conditions by improving its domain-invariance. However, all these state-of-the-art approaches require training samples from multiple domains with corresponding domain-labels for enforcing the domain-invariance. Hence, these approaches [11], [12], [13], [14] cannot be used in low-resource conditions in which the training dataset contains all samples from only one domain, lacking diversity. To address this issue, a ''within-sample similarity loss'' (WSSL) based approach was proposed in [10] and [15]. However, like the other approaches used for improving the domain-invariance ([11], [12], [13], [14]), the WSSL also does not explicitly encourage the network to learn class-discriminative embeddings, which is essential to boost the performance in case of high interclass similarities and intraclass variations.

Several previous studies have shown that, learning class-discriminative embeddings improve the robustness of the system to high interclass similarities and high intraclass variations [16], [17], [18], [19], [20]. In general, they use an auxiliary loss during the training to increase interclass sample distances and reduce intraclass sample distances in the embedding space.

Motivated by all these, in this work, we propose a centroid similarity loss (CSL) with adaptive gradient blending (AGB) (denoted as *CSL-with-AGB*) strategy to improve the robustness of a LID system to unseen target domain conditions. Unlike most of the existing approaches ([11], [12], [13], [14]), the proposed *CSL-with-AGB* can improve the generalization of the system even in low-resource conditions, as it does not require training samples from multiple domains with corresponding domain labels.

Specifically, in the proposed approach, the network first performs a bi-resolution processing of the speech using a set of two separate embedding extractors (which can be treated as two branches of the network). Such analysis allows them to encode complementary contents in the input, which in turn improves the generalization of the system to some extent [15]. We then propose to use the centroid similarity loss (CSL) to further improve the generalization of the system. Specifically, the CSL encourages the network to minimise intraclass variations by reducing distance between embedding of a given training sample and its class-centroid (mean embedding), and simultaneously minimise the inter-class similarities by suppressing similarities between the embedding and centroids of all other language classes. Since the interclass similarities suppressed by the CSL includes the similarities due to language-specific contents, as well as those due to domain-specific contents, this approach improves both discriminative power and domain-invariance of the embeddings. In our approach, the CSL is applied separately on both branches of the network to improve their discriminative ability.

However, application of CSL can sometimes force the two branches of the network to learn at significantly different rates, which results in an unbalanced learning from them. Such unbalanced learning diminishes their ability to encode complementary contents in the input and results in degraded performance of the system in unseen target domain conditions. To address this issue, we propose to include the adaptive gradient blending (AGB) [21], [22] with the CSL. Specifically, with the help of a set of auxiliary classifiers attached to the two embedding extractors, the AGB monitors their learning behavior and produces a set of dynamic weights to combine the auxiliary losses from the two branches with the primary loss. Such dynamic fusion of auxiliary losses guides the two branches to have a balanced learning, leading to improved robustness to unseen target domain conditions.

Our CSL based approach is motivated by the previous works in [20], [23], [24], [25], and [26]. Specifically, the CSL is similar to the work in [26], in which a class-wise centroid distance metric learning has been used to improve the discriminative ability of a deep neural network (DNN) based acoustic event detector (AED). However, in our case, we use CSL to improve the robustness of a LID system to real-world conditions, where, high interclass similarities, high intraclass variations and domain-mismatches are expected. To the best of our knowledge, this work is the first one to explore CSL for LID task. Furthermore, unlike the prevailing approaches [20], [23], [24], [25], [26], we propose to use a bi-resolution processing of the input along with adaptive gradient blending (AGB) to improve the robustness of the system to unseen target domain conditions.

III. CONTRIBUTIONS

The techniques presented in this work are aimed at improving the generalization of a LID system to unseen target domain conditions when the training dataset used to train the system contains very limited diversity. Highlights of this work can be listed as follows:

1) A CSL-based strategy used along with bi-resolution processing of the speech for LID.
2) An AGB-based strategy to enhance the ability of the two branches of the network to encode complementary contents, leading to improved generalization.
3) Extensive experimentation using two different speech corpora, which contain different level of domain-diversity.

Rest of this paper is organised as follows. In Section II, we first give details of the LID system which performs a bi-resolution processing of the speech with the help of two separate input branches, followed by details about the proposed CSL. In Section III, we discuss the problem of unbalanced learning from the two branches of the network and a AGB-based solution to address it. The Section IV gives

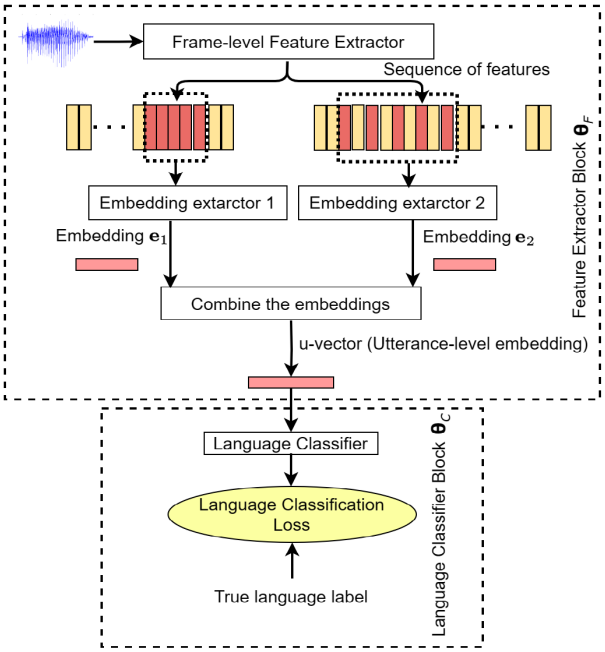

FIGURE 1. Block diagram of the bi-resolution processing based LID system. Red coloured frames in sequence of feature vectors indicate the frames selected as input within a given chunk.

details about the datasets used in the study. Section V contains details about experiments and results, followed by conclusion in Section VI.

IV. CSL-BASED APPROACH TO IMPROVE THE GENERALIZATION OF LID SYSTEM

As the first step in the proposed CSL-with-AGB strategy, the network performs a bi-resolution processing of the input speech sample. The block diagram of the network performing bi-resolution processing of the speech is shown in Figure 1. It contains a feature extractor block (with trainable parameters $\theta_{\boldsymbol{F}}$) to produce an utterance-level embedding (u-vector) [10], [15] of the speech, followed by a language classifier block (with parameters $\theta_{\boldsymbol{C}}$) to predict the language label. The feature extractor block contains a pre-trained bottleneck feature (BNF) extractor [27] at the front-end to convert the input speech into a sequence of BNF features. This sequence of BNFs is then analysed by a set of two utterance-level embedding extractors to get two intermediate-level embeddings (denoted as $\mathbf{e}_1$ and $\mathbf{e}_2$ in Figure 1).

Both embedding extractors have identical architecture. They first process the sequence of BNF vectors by dividing it into fixed-length chunks to produce LID-seq-senones [28],[1] which are then combined into utterance-level embedding (denoted as $\mathbf{e}_1$ or $\mathbf{e}_2$ in Figure 1) [10]. Unlike the traditional LID systems [29], [30], [31] which analyse the input at single temporal resolution, the two embedding extractors in our network are designed to process the input at two different temporal resolutions [10], [15]. Specifically, the embedding extractor-1 processes the input by dividing it into chunks of T_{c1} seconds, and embedding extractor-2 processes the same input using chunks of T_{c2} seconds. Furthermore, while the embedding extractor-1 considers all BNF vectors as input within the chunk of T_{c1} seconds, the embedding extractor-2 considers only alternate BNFs as input, within the chunk of T_{c2} seconds. The motivation for such kind of analysis is the following. We visualize the given speech sample as a combination of two components: a fast-varying component related to the foreground speech, and a constant (or slowly-varying) component related to the background (related to speaker, channel, etc., that remain constant within a given utterance) [32], [33]. Analysing the given speech at two different temporal resolutions allows the two embedding extractors to capture dissimilar contents about the foreground (due to its fast-changing nature) and similar contents about the background (as it remains almost constant) [10], [15].

The output of these two embedding extractors ($\mathbf{e}_1$ and $\mathbf{e}_2$) are then combined together into the final utterance-level representation called u-vector. As in [10], we use a self-attention based fusion of embeddings $\mathbf{e}_1$ and $\mathbf{e}_2$ to get the u-vector. Such arrangement allows the u-vector to gather complementary LID-specific contents in the speech, obtained by processing the speech in two different temporal resolutions.

This u-vector is then fed to the language classifier ($\theta_{\boldsymbol{C}}$) to form an end-to-end LID network (as in Figure 1). Let N_l be the number of nodes in the output layer of this classifier with *softmax* activation. For an input sample $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots\ldots, \mathbf{x}_T)$, with T being the length of the input sequence of feature vectors, let the output of the language classifier be $P(\hat{y}_l|\mathbf{X}, \theta_{\boldsymbol{F}}, \theta_{\boldsymbol{C}})$; where $\hat{y}_l \in \{l_1, l_2, \ldots, l_{N_l}\}$ denote the predicted language classes. Let y_l denote the true language label in 1-hot encoded format. For the given training sample $\mathbf{X}$, we compute the cross entropy loss[2] as:

$$L_p(\theta_{\boldsymbol{F}}, \theta_{\boldsymbol{C}}) = -\sum_{l=1}^{N_l} y_l \log(P(\hat{y}_l|\mathbf{X}, \theta_{\boldsymbol{F}}, \theta_{\boldsymbol{C}})) \qquad (1)$$

We denote this end-to-end LID network as *2Arm-u-vec-Net*, which is trained using the cross entropy loss in Eq.1. Note that, while such bi-resolution processing allows the network to gather dissimilar contents in the input, there is no explicit encouragement on the network to improve discriminative power and domain-invariance of the embeddings from the two input branches. Hence, this approach can provide only a limited improvement in the generalization. In order to address this limitations, we use the CSL.

A. CENTROID SIMILARITY LOSS

The proposed centroid similarity loss (CSL) is a distance metric learning approach, which reduces the distance between

[1]These LID-seq-senones are intermediate-level LID-specific features, which compactly represent the contents in the given chunk of speech.

[2]The symbol p used as a suffix indicates that this is the primary loss used in the training.

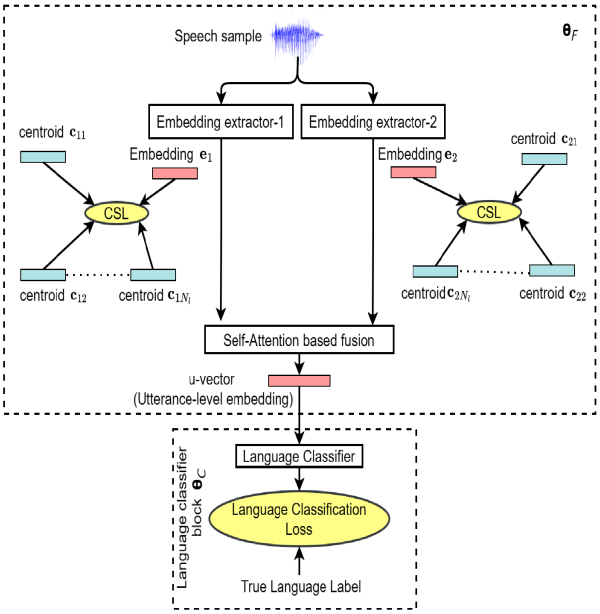

FIGURE 2. Block diagram of the proposed CSL based approach. This network is denoted as *Lnet_CSL*.

an embedding and its class centroid and simultaneously increases the distance between the embedding and centroids of other classes. Thus, it forces the network to minimize intraclass variations and interclass similarities in the embedding space. Figure 2 shows the block diagram of the proposed CSL approach.

In the proposed approach, the CSL is applied separately on both the embedding extractors of the network to improve their discriminative power. In this approach, a normalized similarity score for both the embeddings of a given training sample is first computed as follows.

$$S_{nl}(\theta_{\boldsymbol{F1}}) = \frac{exp(S_c(\mathbf{e}_n, \mathbf{c}_{nl}))}{\sum_{m=1}^{N_l} exp(S_c(\mathbf{e}_n, \mathbf{c}_{nm}))}$$
$$\text{where} \quad n \in \{1, 2\}, \quad l = 1, 2, \ldots, N_l \tag{2}$$

where $\mathbf{c}_{nl}$ represents the centroid (mean) of embeddings in l^{th} language for the n^{th} embedding extractor, N_l represents the total number of classes, S_c is a similarity measure between the n^{th} embedding $\mathbf{e}_n$ and a given centroid, and $\theta_{\boldsymbol{F1}}$ represents parameters of the feature extractor block of the network, excluding the parameters of the attention network used for fusion of the embeddings. We use the cosine similarity for obtaining S_c in this work. Note that, ideally, an embedding of a given training sample should have high cosine similarity with corresponding class centroid, and should have low similarities with centroids of other language classes. Hence, in such situations, the normalized similarity score $S_{nl}(\theta_{F1})$ will have a value close to "1" for intraclass similarity and will have values close to "0" for other interclass similarities. When the given training sample has high intraclass variation, and/or high interclass similarities, the value of $S_{nl}(\theta_{F1})$ will be significantly less that "1" for intraclass similarity, and will be significantly higher than "0" for other interclass similarities. Using this normalized similarity score, the CSL is computed as follows:

$$L_{csl}(\theta_{\boldsymbol{F1}}) = -\sum_{l=1}^{N_l} y_l \log(S_{1l}) - \sum_{l=1}^{N_l} y_l \log(S_{2l}) \tag{3}$$

where y_l indicates the true label of the given training sample. With the CSL as an auxiliary loss, the LID network (denoted as *Lnet_CSL*) is trained using the following total loss function.

$$L_T(\theta_{\boldsymbol{F}}, \theta_{\boldsymbol{C}}) = L_p(\theta_{\boldsymbol{F}}, \theta_{\boldsymbol{C}}) + \alpha_{csl} L_{csl}(\theta_{\boldsymbol{F1}}) \tag{4}$$

where $L_T(\theta_F, \theta_C)$ is the total loss, $L_p(\theta_F, \theta_C)$ is the primary language classification (cross-entropy) loss, and $L_{csl}(\theta_{F1})$ is the CSL. The scalar value α_{csl} represents the trade-off parameter between the primary loss and the CSL. Note that, when a given training sample has high intraclass similarity and low interclass similarities (which should be the case ideally), the magnitude of CSL will be small, indicating that there will be less/no penalty on the network. When a training sample has high intraclass variation and/or high interclass similarities, the magnitude of CSL will be large, indicating that there will be more penalty on the network. Due to this combination of CSL with the primary classification loss $L_p(\theta_F, \theta_C)$, the embeddings from both embedding extractors ($\mathbf{e}_1$ and $\mathbf{e}_2$) ideally become more discriminative. This in turn improves the discriminative capabilities of the u-vector, which is obtained by combining the two embeddings.

Note that, as the utterance-level embeddings $\mathbf{e}_1$ and $\mathbf{e}_2$ of a given speech sample carry both language-specific and domain-specific contents encoded in them, the interclass similarities considered by the CSL includes the similarities due to both language-specific and domain-specific contents. Hence, apart from improving the discriminative power of the embeddings, the CSL encourages the LID network to suppress the domain-specific contents to some extent.

B. THE PROBLEM OF UNBALANCED LEARNING

Note that, the proposed CSL is applied independently on the two embedding extractors of the network, and there is no direct interaction between them. Since the CSL continuously motivates the two embedding extractors to learn more and more discriminative contents, and, due to the fact that they process the input at different temporal resolutions, application of CSL can sometimes force the two branches (embedding extractors) of the network to learn/converge at different rates. Sometimes, this difference in the rate of learning can be very significant. Due to this, at the end of training process, only one branch of the network learns language-discriminative contents adequately, while the other branch settles to a sub-optimal set of parameters.

Figure 3 demonstrates one such situation. In this Figure, we have plotted the training and validation losses for the primary language classifier and the two embedding

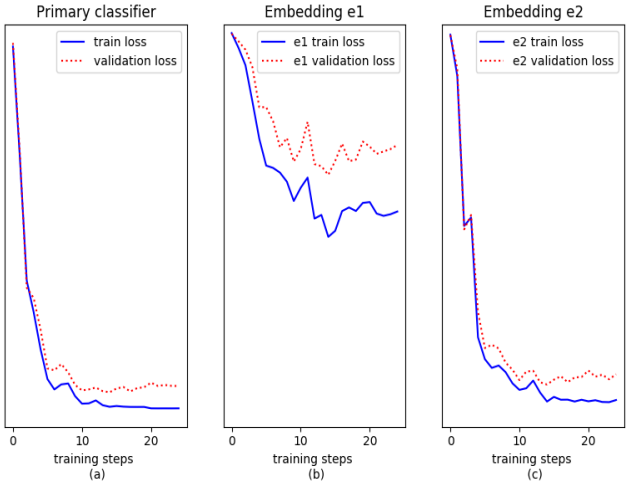

FIGURE 3. Plot showing variation in training and validation losses, computed on the (a) primary classifier, (b) embedding extractor-1 and (c) embedding extractor-2, of CSL-based network. It is seen that, the two embedding extractors learn at different rates, and only one of them learns adequately (embedding extractor-2).

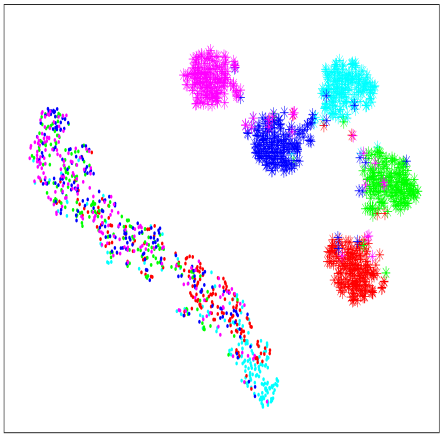

FIGURE 4. t-SNE plot showing the effect of having unequal learning by two embedding extractors of *Lnet_CSL*. Clusters from $\mathbf{e}_2$ (denoted using $*$ symbol) have very clear separation, whereas, clusters from $\mathbf{e}_1$ (denoted using $\bullet$ symbol) have high overlap.

extractors[3] of the *Lnet_CSL*. It is seen that, the embedding extractor-2 of the network learns at a faster-rate and efficiently encodes the language-discriminative contents. However, the embedding extractor-1 learns at a significantly slower pace. In such situations, the self-attention network starts giving more importance to the fast-learning branch and ignores the slow-learning branch, as it helps the network to minimise the training loss. As a result, only fast-learning branch of the network gets sufficient encouragement to learn discriminative contents and moves quickly towards the saturation (beyond which there is no reduction in the training loss), leaving the other branch almost unattended. Due to this, at the end of training process, only fast-learning branch learns language-discriminative contents adequately, while the other branch settles to a sub-optimal set of parameters.

The t-SNE plot in Figure 4 provides some more insight on this issue. In this figure, we have plotted the embeddings $\mathbf{e}_1$ and $\mathbf{e}_2$ of the CSL network (*Lnet_CSL*). These samples belong to five different languages which are shown in different colours. In this plot, it is seen that, the embeddings $\mathbf{e}_2$ (denoted using $*$ symbol) have formed very distinct and compact clusters. But, the $\mathbf{e}_1$ (denoted using $\bullet$ symbol) have formed overlapping clusters with relatively lesser compactness, indicating that they have relatively less discriminative power.

Such unbalanced learning from the two branches of the network limits its ability to encode complementary LID-specific contents in the input, as it requires both embedding extractors to learn language-specific contents adequately. Because of this, the generalization of the LID system degrades to some extent.

In such situations, we might naively think that continuing the training process to few more epochs will force the slow-learning branch of the network to learn sufficient language-discriminative contents. However, once the fast-learning branch reaches the saturation (which gives a reasonable performance on the validation data), any further efforts to continue the training process results only in overfitting of that branch to the training data, and it does not actually encourage the learning of other (slow-learning) branch. Hence, in order to encourage both branches of the network to learn in a balanced way, we need a sophisticated training strategy, which is presented in the next section.

V. AGB FOR BALANCED LEARNING

To address the issue of unbalanced learning, we use the adaptive gradient blending (AGB) strategy. This AGB strategy is motivated by the previous works in [21], [22], and [34], where, a multi-view learning approach is used to learn complementary contents from different modes of input. For example, the ''XSleepNet'' used in [22] for automatic sleep staging, contains two subnetworks in the front-end to analyse the raw input signal and its time-frequency representation. Similarly, the network used in [21] (for audio and music classification), contains four subnetworks at the front-end to analyse the input in Mel-scale, Constant-Q transform spectrogram, Gammatone, and raw signal forms respectively.

Unlike these approaches where multi-modal inputs are used for obtaining complementary contents in the input [21], [22], [34], our network processes the same input at two different temporal resolutions. To the best of our knowledge, AGB based strategy has been not used previously for improving the robustness of a LID network to unseen target domains.

[3]Note that, the embedding extractors in the original CSL network (Figure 2) does not have language classifiers directly attached to them. Hence, after every training step, we approximate the losses on these two embedding extractors by feeding their output directly to the primary language classifier (without combining the two embeddings into u-vector).

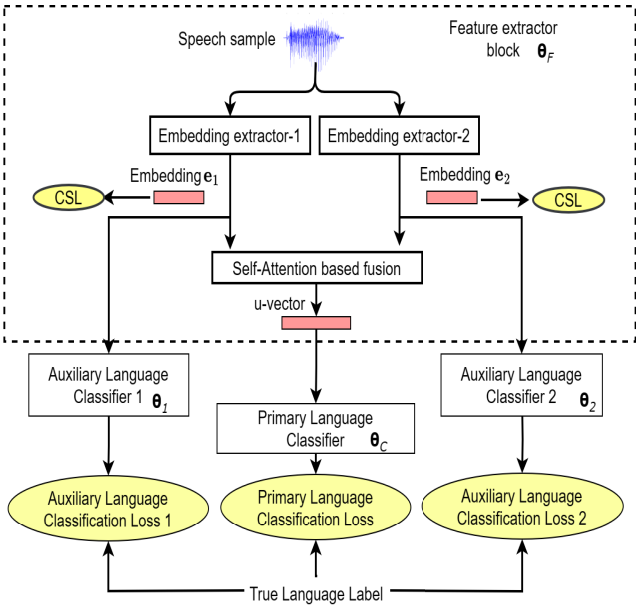

FIGURE 5. Block diagram of the proposed adaptive gradient blending (AGB) approach. In addition to the primary language classifier, AGB uses two auxiliary classifiers to monitor the learning behaviour of the embedding extractors.

The block diagram of the proposed CSL with AGB approach is shown in Figure 5. Compared to the original CSL-based network shown in Figure 2, the CSL with AGB approach contains two additional classifiers to monitor the learning of two embedding extractors. During the training process, the AGB dynamically assigns different weights to the two embedding extractors (and primary classifier) of the network depending on their learning behaviour, so that both branches learn to encode language-discriminative contents adequately.

Let L_1 denote the auxiliary language classification loss, computed using the output of the auxiliary classifier attached to the embedding extractor-1. With $\hat{y_{c1}} \in \{l_1, l_2, \ldots, l_{N_l}\}$ as the predicted language classes, L_1 is computed as:

$$L_1(\theta_{\boldsymbol{F}}, \theta_{\mathbf{1}}) = -\sum_{l=1}^{N_l} y_l \log(P(\hat{y_{c1}}|\mathbf{X}, \theta_{\boldsymbol{F}}, \theta_{\mathbf{1}})) \tag{5}$$

where $\theta_{\mathbf{1}}$ indicates the parameters of the auxiliary classifier 1.

Let L_2 denote the auxiliary language classification loss, computed using the output of the auxiliary classifier attached to the embedding extractor-2. With $\hat{y_{c2}} \in \{l_1, l_2, \ldots, l_{N_l}\}$ as the predicted classes, L_2 is computed as:

$$L_2(\theta_{\boldsymbol{F}}, \theta_{\mathbf{2}}) = -\sum_{l=1}^{N_l} y_l \log(P(\hat{y_{c2}}|\mathbf{X}, \theta_{\boldsymbol{F}}, \theta_{\mathbf{2}})) \tag{6}$$

where $\theta_{\mathbf{2}}$ indicates the parameters of the auxiliary classifier 2.

Using the equations Eq.1, Eq.5 and Eq.6, we first compute the training loss and testing (approximated using validation data) loss for the primary and two auxiliary classifiers, which are then used to monitor their learning behaviour.

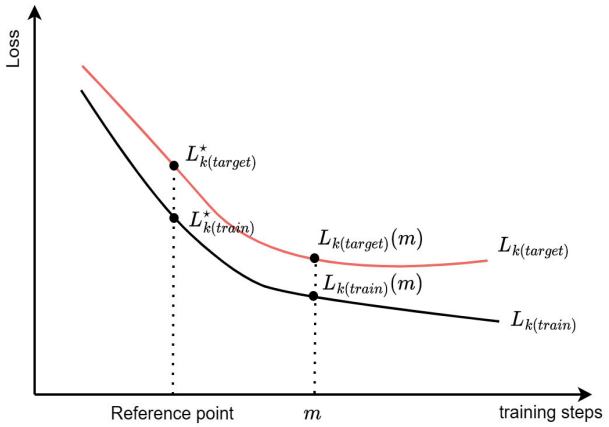

FIGURE 6. General plot showing variation of training and target losses with respect to number of training steps.

We monitor the learning behaviour of classifiers in k^{th} branch of the network, $k \in \{1, 2, p\}$,[4] using generalization parameter (G_k) and overfitting parameter (O_k) [21]. In general, G_k represents the information gained upon the target domain through training process, and O_k represents the gap between information gained on the training domain and the target domain. For the given training step m, the generalization and overfitting parameters can be estimated as [21]:

$$G_k(m) \approx L^{\star}_{k(target)} - L_{k(target)}(m) \;\; k \in \{1, 2, p\} \tag{7}$$

$$O_k(m) \approx (L^{\star}_{k(train)} - L_{k(train)}(m)) - (L^{\star}_{k(target)} - L_{k(target)}(m)) \tag{8}$$

In Eq.7 and Eq.8, $L_{k(train)}(m)$ and $L_{k(target)}(m)$[5] denote respectively the loss computed on training set and loss on the target domain at the m^{th} step. $L^{\star}_{k(train)}$ and $L^{\star}_{k(target)}$ are respectively some reference training loss and reference target loss values.

Figure 6 indicates how training and target losses vary typically with respect to training steps. In general, when a branch (embedding extractor) is near saturation/overfitting, the change in target loss, measured between a reference point and given training step, will be very small compared to change in training loss. Hence, a saturating branch of the network will have large O_k and small G_k values. In contrast, a branch which is away from saturation (which is learning nicely) will have larger G_k and small O_k, as both training and target losses will be decreasing at a similar rate.

Note that, when a branch is generalizing well, both training and target loss values will decrease at a similar rate, due to which, it will have large G_k and small O_k values. In contrast, a branch near its saturation will have large O_k and small G_k

[4]Here, $k = 1$ corresponds to branch-1 (embedding extractor-1), $k = 2$ corresponds to branch-2, and $k = p$ corresponds to the primary language classifier.

[5]Here, (train) and (target) are used just for indicating the domain on which the loss is computed.

values, as only the training loss reduces in this case, but target loss remains almost constant.

Using G_k and O_k, the fusion weights for individual losses are computed as:

$$w_k(m) = \frac{1}{z}\frac{G_k(m)}{O_k^2(m)}, \tag{9}$$

where z is a normalization factor.

During the training process, the total loss is computed by combining the three cross entropy losses (L_p, L_1 and L_2) and the CSL. However, unlike CSL, the three cross entropy losses are combined dynamically. We compute the total loss as follows:

$$L_T(m) = \sum_k w_k(m)L_k(m) + \alpha_{csl}L_{csl}(\theta_{\boldsymbol{F}}), \quad k \in \{1, 2, p\} \tag{10}$$

where L_T is the total loss at training step m, and $w_k(m)$ indicates the weight associated with the k^{th} classifier at that training step. The $w_k(m)$ is estimated dynamically depending on the learning behaviour of k^{th} branch.

The G_k and O_k values estimated using Eq.7 and Eq.8 require a reference training loss and a reference target loss values. As the actual loss curves will be noisy in nature, considering the loss values at a single preceding training step as reference point may lead to sub-optimal results. Hence, we take average of previous r values as the reference for both train loss and target loss [21]. The Algorithm 1 shows the overall process to obtain the weights $w_k(m)$.

The adaptive weights used in Eq. 10 as well as the loss values used in their computation are updated after every mini-batch of training samples. Since the actual loss curves will be noisy in nature, we use a smoothed version of the loss by taking the mean values over a window of length r, as indicated in the Algorithm 1.

Algorithm 1 Adaptive Weight Calculation for m^{th} Training Step

procedure Adaptive weight($L_{train}, L_{target}, L^{\star}_{train}, L^{\star}_{target}, r$)

Input: $L_{train}[1, 2, \ldots., m]$: list of training loss values
$L_{target}[1, 2, \ldots., m]$: list of target loss values
$L^{\star}_{train}$: current best training loss value
$L^{\star}_{target}$: current best target loss value
r: window length for computing mean

Output: $w(m)$: weight at m^{th} step of the training

$\bar{L}_{train}(m) = mean(L_{train}[(m-r)\ldots., m])$
$\bar{L}_{target}(m) = mean(L_{target}[(m-r), \ldots., m])$
$G(m) = L^{\star}_{target} - \bar{L}_{target}(m)$
$O(m) = [L^{\star}_{train} - \bar{L}_{train}(m)] - [L^{\star}_{target} - \bar{L}_{target}(m)]$
$w(m) = \frac{1}{z}(G(m)/O^2(m))$
If $\bar{L}_{train} < L^{\star}_{train}$ **then** $L^{\star}_{train} = \bar{L}_{train}$
If $\bar{L}_{target} < L^{\star}_{target}$ **then** $L^{\star}_{target} = \bar{L}_{target}$

end procedure

VI. DATASETS USED IN THE STUDY

In this study, we used the "IIT-Mandi Indian language dataset"(denoted as IIT-Mandi-DS) [15] and the dataset used in cross-channel LID task of "AP20-OLR Challenge"(AP20-OLR-DS) [35].

A. IIT-MANDI INDIAN LANGUAGES DATASET

This corpus contains two parts: IIT-Mandi Read speech and IIT-Mandi YouTube dataset.[6] As shown in Table 1, there are eight languages in this corpus. Among these eight languages, Assamese, Bengali, Gujarati, Hindi and Odia belong to Indo-Aryan language family. The south Indian languages Kannada, Malayalam and Telugu belong to Dravidian language family [36]. Hence, these languages contain high interclass similarities.

The IIT-Mandi Read speech dataset (denoted as *Read-Speech-DS*) contains audio files obtained from news broadcasts in All India Radio.[7] Each language in this dataset contains around 4.5 hours of speech data from at least 15 speakers. In our experiments, we use around 75% samples from this dataset for training and use remaining for validation.

The IIT-Mandi YouTube dataset (denoted as *YouTube-DS*) contains audio files extracted from various YouTube videos on online teaching, personal interviews, etc. Each language contains samples from at least 10 speakers. Note that, there is significant **domain-mismatch** between IIT-Mandi Read speech and IIT-Mandi YouTube datasets in terms of channel, type of speech, background conditions, etc. In our experiments, we use the YouTube-DS only for testing. The Table 1 shows details like number of hours of speech data, number of utterances and speakers in each language in these two datasets.

B. AP20-OLR CHALLENGE DATASET

Second one is the dataset used for task-1 (cross-channel LID) of AP20-OLR challenge. In this task, the LID system has to handle the test samples with channel-mismatch [35]. The training dataset provided in the challenge includes the dataset used in previous OLR challenges. It contains languages from several language families like Austroasiatic languages (e.g., Vietnamese), Indo-European languages (e.g., Russian), Altaic languages (e.g., Korean, Japanese), etc. [35]. Most of the training dataset is collected using Mobile channel. However, there are some intra-domain variations present in the corpus. For example, the samples for Russian, Korean and Japanese are recorded in two different background condition: quiet and noisy [35], [37]. Furthermore, the training dataset used in AP20-OLR includes the samples used in AP19-OLR cross-channel LID task. Hence, the dataset contains samples from different channels. Due to these differences

[6]available online at https://speechiitmandi.github.io/air/
[7]https://newsonair.gov.in/

TABLE 1. Details about the IIT-Mandi Indian languages corpus used in this work.

Language	Read-Speech-DS			YouTube-DS		
	#Hours	#Utterances	#Spkrs	#Hours	#Utterances	#Spkrs
Assamese	4.3	1850	16	1.0	366	12
Bengali	5.4	1950	15	1.0	361	11
Gujarati	5.1	1835	15	1.0	362	10
Hindi	5.4	1962	18	1.0	367	15
Kannada	4.6	1844	16	1.0	363	12
Malayalam	4.5	1780	15	1.0	362	12
Odia	4.5	1765	15	1.0	363	10
Telugu	5.0	1960	15	1.0	366	11

TABLE 2. Details of the AP20-OLR corpus used in this work.

Language	Train		Test	
	#Hours	#Utterances	#Hours	#Utterances
Russian	19.2	13996	3.5	1800
Japanese	14.2	16919	2.2	2254
Vietnamese	19.1	16128	3.2	1800
Uyghur	23.0	12997	-	-
Kazakh	16.5	10294	-	-
Mandarin	20.4	18563	-	-
Cantonese	20.2	17913	3.1	2394
Korean	15.5	16989	2.2	1800
Indonesian	17.1	15381	3.2	1800
Tibetan	17.2	18871	-	-

in background and channel condition, there are significant intraclass variations in the AP20-OLR dataset.

Note that, the channel conditions of the test samples in cross-channel LID task of AP20-OLR are different than those in the training set (unknown to the user). Hence, there is **channel-mismatch** between the train and test sets. The training dataset contains 10 languages and testing dataset contains 6 languages as shown in Table 2. In our experiments, we use around 80% samples from the training dataset for training and use remaining for validation.

VII. EXPERIMENTAL STUDIES AND RESULTS

In our experiments, we evaluate the performance of our LID systems in **seen** and **unseen** test conditions. Here, seen test set is the one whose domain (channel/background) conditions are seen by the system during the training. In each dataset, we use the respective validation set as the seen test set. Unseen test set is the one whose domain conditions are unseen by the system during the training. It represents the unseen target domain condition for the system, which is common in real-world applications. Specifically, for the systems trained on Read-Speech-DS, we use the YouTube-DS as the unseen test set. For the systems trained on AP20-OLR-DS, the corresponding cross-channel test dataset forms the unseen test set. We have not used any data-augmentation techniques in this work. We have used the BNF features obtained using pre-trained BNF extractor [27]. Note that, this BNF extractor was not explicitly trained to handle domain-mismatches [27], [38].

Performance of the systems are evaluated using two different metrics: Accuracy (%) and C_{avg} (%) metric used in NIST LRE evaluations [39]. Lower value of C_{avg} indicates better performance. In each case, we run M trials ($M = 10$) and take the average of obtained performances.

We first conduct the experiments on seen test sets (validation datasets) to show the effectiveness of proposed approaches in domain-matched conditions. Followed by this, we conduct experiments on unseen test sets which represent the real-world scenario.

A. BASELINE SYSTEM

We use the state-of-the-art x-vector based LID system [29] as our baseline. The x-vector network uses time delay neural network (TDNN) architecture at the front-end, followed by a statistics pooling layer and a classification network. Specifically, the network contains 5 TDNN layers, with each layer having 512 nodes. The output of last TDNN layer is processed using statistics pooling layer, to obtain an utterance-level representation. This utterance-level representation is then processed by a set of two dense layers having 512 nodes. Final dense classification layer has N_l (which equals the number of languages in the dataset) nodes. We train the x-vector network using the cross entropy loss function. We used Adam optimizer with 0.0001 as the learning rate. We then use the pre-trained x-vector network as an embedding extractor and use it to extract x-vectors of all speech samples. These x-vectors are then processed by a separate back-end classifier. Specifically, we use a logistic regression classifier as suggested in [35]. Performance of this system (denoted as *x-vector-LR*) is given in 1^{st} row of Table 3.

B. PERFORMANCE OF BI-RESOLUTION PROCESSING BASED APPROACH

Here, we give the performance of the network which performs bi-resolution processing of the input speech sample (denoted as *2Arm-u-vec-Net*). As shown in Figure 1, this network contains a set of two identical embedding extractors. Each embedding extractor contains a set of BLSTM layers to process the input sequence of BNF vectors, by diving it into a set of fixed-length chunks. The first embedding extractor analyzes the input by dividing it into chunks of 0.61 sec (T_{c1}), whereas, the second embeddding extractor uses 0.91 sec (T_{c2}) chunks. The output of BLSTM layers, denoted as LID-seq-senones, are then combined into a fixed-length utterance-level embedding (denoted as $\mathbf{e_1}$ or $\mathbf{e_2}$ in

TABLE 3. Performance in accuracy (*Acc*) and C_{avg} (both given in %) of LID systems in seen test sets.

LID system	Read-Speech-DS		AP20-OLR-DS	
	Acc	Cavg	Acc	Cavg
x-vector-LR	84.80	9.55	93.45	4.05
1Arm-u-vec-Net	88.15	7.17	95.03	3.15
2Arm-u-vec-Net	90.10	6.16	96.26	1.92
2Arm-u-vec-Net+AGB	90.22	6.14	96.30	1.90
Lnet_CSL	93.95	4.84	97.90	1.68
Lnet_CSL+AGB	94.45	4.14	98.16	1.34

Figure 1) using a statistics pooling layer [10]. In case of Read-Speech-DS, the embedding extractor contains set of two BLSTM layers, with 256 and 32 nodes respectively in first and second layers. In case of AP20-OLR-DS, the embedding extractor contains a set of three BLSTM layers, with 384, 256 and 64 nodes respectively in first, second and third layers. The embeddings $\mathbf{e_1}$ and $\mathbf{e_2}$ are then combined together into u-vector, which is then given to the output layer. The output layer has N_l number of nodes with *softmax* activation.

Performance of *2Arm-u-vec-Net* is given in 3^{rd} row of Table 3. It is seen that, compared to baseline x-vector based system, the proposed bi-resolution processing based approach gives better performance. This is because, unlike the x-vector system which contains a single embedding extractor, the 2Arm-u-vec-Net contains a set of two embedding extractors operating at two different temporal resolutions. Such arrangement allows the network to gather dissimilar (which is complementary to some extent) information about the speech, leading to better performance.

The 2^{nd} row of Table 3 shows the performance of a LID network that uses same number of BLSTM nodes as in *2Arm-u-vec-Net*, but does not use bi-resolution processing. In this case, the network contains only one embedding extractor (denoted as *1Arm-u-vec-Net*). The network used for Read-Speech-DS contains set of 2 BLSTM layers with 512 and 64 nodes respectively in first and second layers, and the network for AP20-OLR-DS contains 3 layers with 768, 512 and 128 nodes respectively in first, second and third layers. Chunk length of 0.61 sec is used in this case. From the results, is seen that, the *2Arm-u-vec-Net* performs better than *1Arm-u-vec-Net*. This indicates that bi-resolution processing indeed leads to better performance.

C. PERFORMANCE OF PROPOSED CSL-BASED APPROACH

In this case, we include the proposed CSL in the training process and the network (denoted as *Lnet_CSL*) is trained according to Eq. 4. The value of the trade-off parameter α_{csl} is set empirically as 0.20. Note that, we train the CSL network using only primary language classification loss for first training epoch. This makes sure that the network learns LID-specific contents to some extent before the computation of first set of language centroids. (Without this, the initial set of centroids would be computed with randomly initialized weights, which will not contain any language-specific information encoded in them. Computing similarities with such centroids using CSL will adversely effect the learning). For all subsequent epochs, we use a two step procedure. First, the centroids for all languages are updated using training samples in the given mini-batch. During this step, the model parameters (obtained after previous mini-batch) are kept unchanged. In the second step, parameters of the model are updated by using the loss function in Eq. 4. This two step procedure enables the network to consider the interclass and intraclass similarities efficiently during the parameter optimization.

The performance of *Lnet_CSL* is given in 5^{th} row of Table 3. It is seen that, the inclusion of CSL in the training process has improved the performance of the network. As, the CSL minimises the interclass similarities and the intraclass variations in the embedding space, the network produces discriminative embeddings, leading to better performance.

The box-plots in Figure 8 indicates how the performance of *Lnet_CSL* varies when experimented with multiple ($M = 10$) trials. The variation in performances of *2Arm-u-vec-Net* is also given for comparison.

It is seen that, compared to the *2Arm-u-vec-Net*, there is significant variation in the performance of *Lnet_CSL*. This is because, the CSL-based network has additional constraint on the embedding extractors to learn discriminative embeddings. We believe that, such additional constraint sometimes leads to unbalanced learning from the two branches, resulting in significant variation in performance. In order to motivate the two branches to have a balanced learning, we include the proposed AGB in the training process.

D. PERFORMANCE OF PROPOSED CSL-WITH-AGB APPROACH

In this case, the network contains two auxiliary classifiers attached to the two embedding extractors. Like the primary classifier, each auxiliary classifier is a simple dense layer with N_l nodes, having *softmax* activation. The whole network is trained according to Eq. 10. The value of the normalization factor z is set experimentally as 1.0, and the length of the window (r) is set as 4. The 6^{th} row of Table 3 summarises the performance of CSL-network with the adaptive gradient blending (denoted as *Lnet_CSL+AGB*). It is seen that, due to inclusion of AGB, the CSL network provides better performance compared to *Lnet_CSL*. This is because, the AGB encourages both embedding extractors of the network to learn language-discriminative contents in a balanced way, which enables them to efficiently encode complementary contents in the input. For comparison, the performance of *2Arm-u-vec-Net* trained with AGB (denoted as *2Arm-u-vec-Net+AGB*) is also given in 4^{th} row of the Table. Compared to *2Arm-u-vec-Net*, the *2Arm-u-vec-Net+AGB* has given only slight improvement in the performance. This is because, unlike the CSL-based network, the *2Arm-u-vec-Net* does not have any explicit constraint on the embedding extractors. Hence, both branches of the network learn

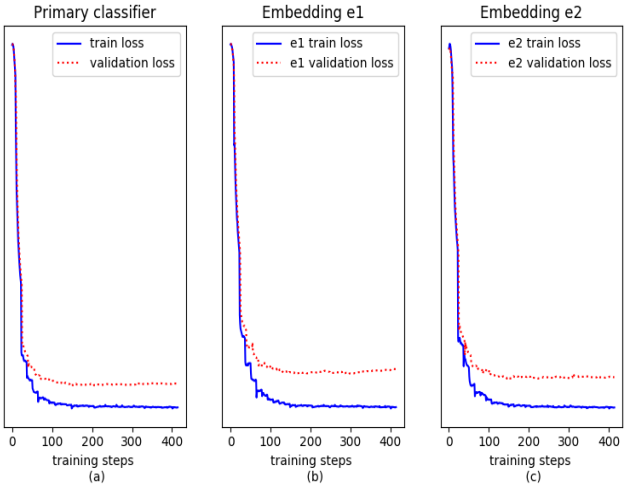

FIGURE 7. Plot showing variation in training and validation losses for (a) primary classifier, (b) auxiliary classifier on branch-1, and (c) branch-2 of *Lnet_CSL+AGB* trained on Read-Speech-DS.

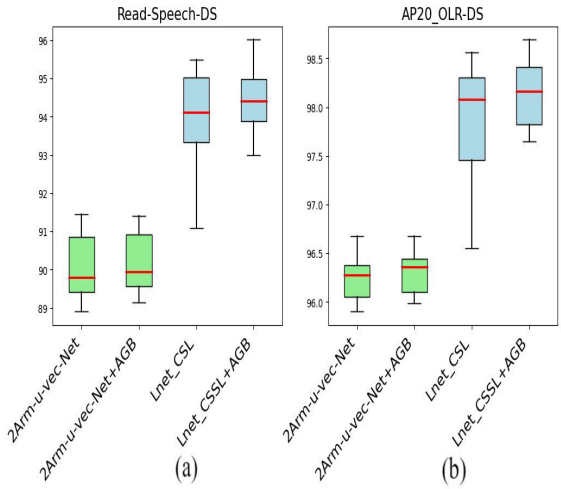

FIGURE 8. Box-plots indicating the variation in the performance of different LID systems trained with and without the AGB strategy. Figure (a) corresponds to systems trained on Read-Speech-DS, and (b) corresponds to AP20-OLR-DS.

adequate language-discriminative contents in a balanced way, even without the AGB.

The loss curves in Figure 7 shows some evidence for the effectiveness of AGB on the CSL-based network. This Figure shows the variation in training and validation losses for the primary classifier and the two auxiliary classifiers (attached to the two embedding extractors) of the network (as shown in Figure 5). It is seen that, training and validation losses on all classifiers reduce at a similar rate and reach the saturation. Due to such balanced learning, both branches of the network learn language-discriminative contents to adequate level, which enable them to encode complementary contents in the speech. This helps the network to perform better.

The box-plots in Figure 8 provide some more evidence to indicate the effectiveness of proposed AGB strategy. These box plots indicate how the performance of systems trained with AGB, are distributed for multiple trails (number of trails, $M = 10$). Here, plot (a) corresponds to the systems trained on Read-Speech-DS and plot (b) corresponds to AP20-OLR-DS. The variation in performance of systems trained without AGB are also given for the comparison. Compared to the variation in performance of CSL based systems trained without AGB, the systems with AGB (*Lnet_CSL+AGB*) have significantly less variation in the performance. Since the AGB guides both embedding extractors to learn in a balanced way, the performance of the network becomes more stable.

The impact of CSL (with AGB) on the output of two embedding extractors can be visualized with the help of 2-dimensional scatter plots in Figure 9 and Figure 10. Here, Figure 9 and Figure 10 respectively correspond to the t-SNE plot of embeddings $\mathbf{e_1}$ and $\mathbf{e_2}$, collected using *2Arm-u-vec-Net* and *Lnet_CSL+AGB*, trained on Read-Speech-DS (embeddings $\mathbf{e_1}$ are shown using $\bullet$ symbol and $\mathbf{e_2}$ using $*$ symbol). We have used samples from seen test set

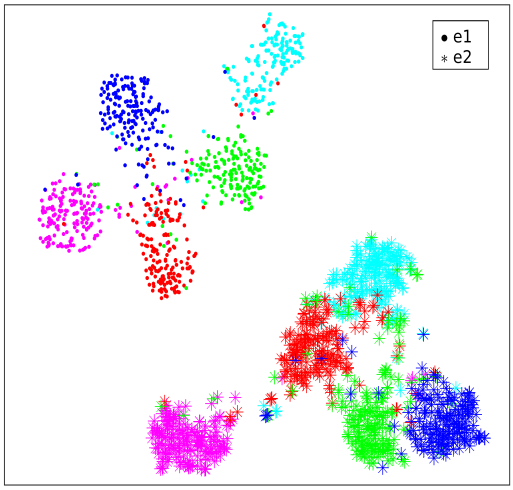

FIGURE 9. t-SNE plot of embeddings $\mathbf{e_1}$ and $\mathbf{e_2}$, obtained from *2Arm-u-vec-Net*, trained on Read-Speech-DS. Here, $\mathbf{e_1}$ and $\mathbf{e_2}$ are represented using separate shapes. Samples from five languages from the corresponding seen test set are used in the plot, which are shown in different colours.

for obtaining these embeddings. These samples belong to five different languages (Assamese represented in Magenta color, Bengali in Cyan, Gujarati in Red, Hindi in Green and Kannada in Blue). It is seen that, compared to the language clusters of *2Arm-u-vec-Net* (Figure 9), the clusters of *Lnet_CSL+AGB* (Figure 10) are better separated from each other. This is because, the CSL (with AGB) encourages network to produce discriminative embeddings.

E. EXPERIMENTS IN UNSEEN TEST SETS

Here, we use test sets containing unseen type of domain-conditions. Obtained results are given in Table 4. Compared to the performance on seen test sets (given in Table 3), performance of all LID systems have reduced significantly in

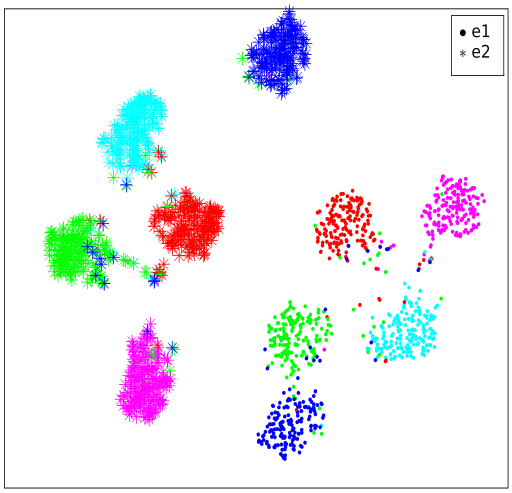

FIGURE 10. t-SNE plot of embeddings e_1 and e_2, obtained from *Lnet_CSL+AGB*, trained on Read-Speech-DS. Compared to the clusters in Figure 9, the clusters in this plot are better separated due to CSL.

TABLE 4. Performance in accuracy (*Acc*) and C_{avg} (both given in %) of LID systems in unseen test sets.

Training dataset	Read-Speech-DS		AP20-OLR-DS	
Testing dataset	YouTube-DS		Cross-channel testset	
	Acc	Cavg	Acc	Cavg
x-vector-LR	63.10	21.50	68.65	14.95
1Arm-u-vec-Net	64.95	19.82	71.50	12.10
2Arm-u-vec-Net	66.16	18.88	72.87	11.01
2Arm-u-vec-Net+AGB	66.20	18.76	73.08	10.86
Lnet_CSL	67.88	16.70	73.15	10.90
Lnet_CSL+AGB	68.35	16.14	73.70	10.32

unseen test sets. This drop in the performance demonstrates the challenges faced by the LID system in unseen target domain conditions.

The 5^{th} and 6^{th} rows of Table 4 respectively show the results obtained for the *Lnet_CSL* and *Lnet_CSL+AGB*. Results of the baseline and *2Arm-u-vec-Net* systems, are also given in the same Table for the comparison. As in the case of seen test set, the *2Arm-u-vec-Net* system has provided slightly better performance than the *x-vector-LR* system and *1Arm-u-vec-Net*. Both *Lnet_CSL* and *Lnet_CSL+AGB* networks have provided reasonable improvement over the baseline, where the later has given the better results. As the proposed CSL encourages the network to minimise intraclass variations and interclass similarities, it increases the ability of the network to discriminate closely related languages. Furthermore, it also improves the domain-invariance to some extent. Like in the case of seen test sets, the inclusion of AGB has improved the overall performance of the systems in unseen target domain conditions. As mentioned earlier, the AGB encourages both branches to learn discriminative contents in the input efficiently, due to which, the network becomes more robust to unseen target domain conditions.

F. COMPARISON OF EFFECTIVENESS IN LOW-RESOURCE AND HIGH-RESOURCE CONDITIONS

In this section, we study how the proposed methods perform in low-resource conditions when compared to high-resource conditions. While we can assume that the Reed-Speech-DS represents a dataset with low-resource conditions (as it has only around 5 hours of speech in each language) and the AP20-OLR-DS represents high resource conditions, the results obtained on these two datasets are not directly comparable as the background and domain conditions in these two are significantly different.

TABLE 5. Comparison of performance of LID systems in low-resource and high-resource conditions when tested using seen test set.

LID system	AP20-OLR-DS-5hr		AP20-OLR-DS	
	Acc	Cavg	Acc	Cavg
x-vector-LR	86.55	7.91	93.45	4.05
1Arm-u-vec-Net	87.02	7.23	95.03	3.15
2Arm-u-vec-Net	87.10	7.05	96.26	1.92
2Arm-u-vec-Net+AGB	87.15	6.95	96.30	1.90
Lnet_CSL	88.94	5.71	97.90	1.68
Lnet_CSL+AGB	89.19	5.44	98.16	1.34

TABLE 6. Comparison of performance of LID systems in low-resource and high-resource conditions when tested using unseen test set.

Training dataset	AP20-OLR-DS-5hr		AP20-OLR-DS	
Testing dataset	Cross-channel testset		Cross-channel testset	
	Acc	Cavg	Acc	Cavg
x-vector-LR	64.33	18.02	68.65	14.95
1Arm-u-vec-Net	64.97	17.78	71.50	12.10
2Arm-u-vec-Net	65.34	17.52	72.87	11.01
2Arm-u-vec-Net+AGB	65.40	17.47	73.08	10.86
Lnet_CSL	65.82	17.05	73.15	10.90
Lnet_CSL+AGB	66.84	16.12	73.70	10.32

Hence, to have a fair comparison, we considered a subset of AP20-OLR-DS by taking approximately 5 hours of speech in each language for training (denoted as AP20-OLR-DS-5hr). Results obtained in seen test set for different LID systems trained on AP20-OLR-DS-5hr are given in Table 5. Results obtained using the original training dataset (AP20-OLR-DS) are also given in the table for easy comparison. Table 6 shows the results obtained using unseen test set.

It is observed that, all systems trained in low-resource condition (trained on AP20-OLR-DS-5hr) have given inferior performance compared to their counterparts trained with original dataset. This is because, with less training data, the system achieves only limited generalization, which results in inferior performance. The overall trend in the improvement in performance due to inclusion of technique like bi-resolution processing, CSL, and CSL-with-AGB is similar in both low-resource and high-resource conditions. Interestingly, when comparing the performances of *Lnet_CSL* and *Lnet_CSL+AGB* in low-resource and high-resource conditions, the latter has given significant improvement over the former in low resource condition than in high-resource condition. This indicates that the proposed combination of CSL-with-AGB is relatively more effective in low-resource condition than in high-resource condition.

G. LIMITATIONS AND ADVANTAGES OF THE PROPOSED APPROACH

Note that, in spite of applying sophisticated techniques like bi-resolution analysis, CSL and CSL-with-AGB, the amount

of improvement in the performance obtained on unseen test set is very limited. This is mainly because of the different domain conditions present in the test samples that are unseen by the system during the training process. This shows the significance of domain-mismatch in real-world conditions. However, using samples in the training domain only, all the proposed approaches encourage the model to generalize better, which otherwise would not have been possible.

VIII. SUMMARY AND FUTURE WORK

In this work, we first presented a bi-resolution processing based approach to improve the robustness of a LID system to unseen target domain conditions. Unlike the prevailing approaches, the LID network in the bi-resolution processing based approach contains a set of two embedding extractors to process the input at two different temporal resolutions, which allows the network to gather complementary contents in the input. We then applied the centroid similarity loss (CSL) to further improve the robustness of the network. The CSL encourages the network to minimise interclass similarities and intraclass variations in the embedding space, thereby encouraging the system to learn discriminative embeddings. Furthermore, the CSL improves the domain-invariance of the network to some extent. However, application of CSL sometimes leads to unbalanced learning from the two branches of the network. To overcome this issue, we proposed to include the adaptive gradient blending (AGB) strategy in the CSL network. Obtained results indicate that the proposed CSL, along with AGB, improves the performance of LID network in both seen and unseen target domain conditions.

Note that, in the present work, we assumed that we do not have access to the target (test) domain during the training process. Hence, we applied the CSL-with-AGB as a domain generalization technique. However, in some cases, we can have access to samples from the target domain for fine-tuning. In such cases, we can use domain adaptation techniques like supervised domain adaptation or unsupervised domain adaptation using adversarial multi-task learning (AMTL) [6], [7], [8], [9]. We can also combine these techniques with CLS-with-AGB to provide better performance in the target domain. We will consider this as a future work. Furthermore, in the present work, we applied the CSL independently on the two embedding extractors. In future, we would like to come up with a strategy to apply a common CSL for both branches.

REFERENCES

[1] H. Li, B. Ma, and K. A. Lee, "Spoken language recognition: From fundamentals to practice," *Proc. IEEE*, vol. 101, no. 5, pp. 1136–1159, May 2013.

[2] K. V. Mounika, S. Achanta, H. R. Lakshmi, S. V. Gangashetty, and A. K. Vuppala, "An investigation of deep neural network architectures for language recognition in Indian languages," in *Proc. Interspeech*, Sep. 2016, pp. 2930–2933.

[3] L. Mateju, P. Cerva, J. Zdánskỳ, and R. Safarik, "Using deep neural networks for identification of slavic languages from acoustic signal," in *Proc. Interspeech*, Sep. 2018, pp. 1803–1807.

[4] R. Huang, J. H. L. Hansen, and P. Angkititrakul, "Dialect/Accent classification using unrestricted audio," *IEEE Trans. Audio, Speech Language Process.*, vol. 15, no. 2, pp. 453–464, Feb. 2007.

[5] N. B. Chittaragi and S. G. Koolagudi, "Dialect identification using chroma-spectral shape features with ensemble technique," *Comput. Speech Lang.*, vol. 70, Nov. 2021, Art. no. 101230.

[6] M. Mclaren, M. K. Nandwana, D. Castán, and L. Ferrer, "Approaches to multi-domain language recognition," in *Proc. Speaker Lang. Recognit. Workshop (Odyssey)*, Jun. 2018, pp. 90–97.

[7] J. A. V. Lopez, N. Brummer, and N. Dehak, "End-to-end versus embedding neural networks for language recognition in mismatched conditions," in *Proc. Speaker Lang. Recognit. Workshop (Odyssey)*, Jun. 2018, pp. 112–119.

[8] B. M. Abdullah, T. Avgustinova, B. Möbius, and D. Klakow, "Cross-domain adaptation of spoken language identification for related languages: The curious case of slavic languages," in *Proc. Interspeech*, Oct. 2020, pp. 477–481.

[9] Q. Wang, W. Rao, S. Sun, L. Xie, E. S. Chng, and H. Li, "Unsupervised domain adaptation via domain adversarial training for speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 4889–4893.

[10] H. Muralikrishna and A. D. Dileep, "Spoken language identification in unseen channel conditions using modified within-sample similarity loss," *Pattern Recognit. Lett.*, vol. 158, pp. 16–23, Jun. 2022.

[11] S. Shon, A. Ali, and J. Glass, "Domain attentive fusion for end-to-end dialect identification with unknown target domain," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5951–5955.

[12] Z. Meng, Y. Zhao, J. Li, and Y. Gong, "Adversarial speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6216–6220.

[13] Y. Shinohara, "Adversarial multi-task learning of deep neural networks for robust speech recognition," in *Proc. Interspeech*, Sep. 2016, pp. 2369–2372.

[14] Y. Zhao, Z. Zhong, F. Yang, Z. Luo, Y. Lin, S. Li, and N. Sebe, "Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6273–6282.

[15] H. Muralikrishna, 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.

[16] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6738–6746.

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

[18] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Process. Lett.*, vol. 25, no. 7, pp. 926–930, Jul. 2018.

[19] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "CosFace: Large margin cosine loss for deep face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5265–5274.

[20] J. Peng, R. Gu, and Y. Zou, "Deep speaker embedding with long short term centroid learning for text-independent speaker verification," in *Proc. Interspeech*, Oct. 2020, pp. 3246–3250.

[21] H. Phan, H. Le Nguyen, O. Y. Chén, L. Pham, P. Koch, I. McLoughlin, and A. Mertins, "Multi-view audio and music classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 611–615.

[22] H. Phan, O. Y. Chén, M. C. Tran, P. Koch, A. Mertins, and M. De Vos, "XSleepNet: Multi-view sequential model for automatic sleep staging," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5903–5915, Sep. 2022.

[23] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vision. (ECCV)*. Amsterdam, The Netherlands: Springer, 2016, pp. 499–515.

[24] W. Chen, J. Huang, and T. Bocklet, "Length- and noise-aware training techniques for short-utterance speaker recognition," in *Proc. Interspeech*, Oct. 2020, pp. 3835–3839.

[25] A. Nanda, W. Im, K.-S. Choi, and H. S. Yang, "Combined center dispersion loss function for deep facial expression recognition," *Pattern Recognit. Lett.*, vol. 141, pp. 8–15, Jan. 2021.
[26] X. Lu, P. Shen, S. Li, Y. Tsao, and H. Kawai, "Class-wise centroid distance metric learning for acoustic event detection," in *Proc. Interspeech*, Sep. 2019, pp. 3614–3618.
[27] A. Silnova, P. Matejka, O. Glembek, O. Plchot, O. Novotnỳ, F. Grezl, P. Schwarz, L. Burget, and J. Cernocky, "BUT/phonexia bottleneck feature extractor," in *Proc. Odyssey*, Jun. 2018, pp. 283–287.
[28] 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.
[29] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using X-vectors," in *Proc. Odyssey*, Jun. 2018, pp. 105–111.
[30] A. Lozano-Diez, O. Plchot, P. Matejka, and J. Gonzalez-Rodriguez, "DNN based embeddings for language recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5184–5188.
[31] J. Pešán, L. Burget, and J. Černocký, "Sequence summarizing neural networks for spoken language recognition," in *Proc. Interspeech*, Sep. 2016, pp. 3285–3288.
[32] S. Shon, W.-N. Hsu, and J. Glass, "Unsupervised representation learning of speech for dialect identification," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Athens, Greece, Dec. 2018, pp. 105–111.
[33] W.-N. Hsu, Y. Zhang, and J. Glass, "Unsupervised learning of disentangled and interpretable representations from sequential data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1876–1887.
[34] W. Wang, D. Tran, and M. Feiszli, "What makes training multi-modal classification networks hard?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12692–12702.
[35] Z. Li, M. Zhao, Q. Hong, L. Li, Z. Tang, D. Wang, L. Song, and C. Yang, "AP20-OLR challenge: Three tasks and their baselines," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2020, pp. 550–555.
[36] S. Dey, M. Sahidullah, and G. Saha, "An overview of Indian spoken language recognition from machine learning perspective," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 21, no. 6, pp. 1–45, Nov. 2022.
[37] Z. Tang, D. Wang, Y. Chen, and Q. Chen, "AP17-OLR challenge: Data, plan, and baseline," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2017, pp. 749–753.
[38] 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.
[39] H. Zhao, D. Bansé, G. Doddington, C. Greenberg, J. Hernández-Cordero, J. Howard, L. Mason, A. Martin, D. Reynolds, E. Singer, and A. Tong, "Results of the 2015 NIST language recognition evaluation," in *Proc. Interspeech*, Sep. 2016, pp. 3206–3210.

MURALIKRISHNA H 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 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.

SUJEET KUMAR received the B.Tech. degree in computer science and engineering from BPUT, Odisha, India, in 2010, the M.S. (by research) degree in computer networks from the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi (IIT Mandi), India, in 2015, where he is currently pursuing the Ph.D. degree in speech signal processing. He was a Project Engineer and a Project Officer with IIT Mandi. His research interests include machine learning, deep learning, speech signal processing, spoken language recognition, and language diarization.

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. degree in computer science and engineering and the Ph.D. degree 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, where he was an Associate Professor, from 2019 to 2023. He is currently 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. She was an Assistant Professor with the Department of Computer Science and Engineering, NIT Goa, India, from 2013 to 2018, where she has been an Associate Professor, since 2018. Her research interests include deep learning, kernel methods, computer vision, speech processing, content-based information retrieval, and pattern recognition.