SPOKEN LANGUAGE IDENTIFICATION IN UNSEEN TARGET DOMAIN USING WITHIN-SAMPLE SIMILARITY LOSS

Muralikrishna H, Shantanu Kapoor, Dileep Aroor Dinesh, Padmanabhan Rajan

MANAS Lab, Indian Institute of Technology Mandi, India

ABSTRACT

State-of-the-art spoken language identification (LID) networks are vulnerable to channel-mismatch that occurs due to the differences in the channels used to obtain the training and testing samples. The effect of channel-mismatch is severe when the training dataset contains very limited channel diversity. One way to address channel-mismatch is by learning a channel-invariant representation of the speech using adversarial multi-task learning (AMTL). But, AMTL approach cannot be used when the training samples do not contain the corresponding channel labels. To address this, we propose an auxiliary within-sample similarity loss (WSSL) which encourages the network to suppress the channel-specific contents in the speech. This does not require any channel labels. Specifically, WSSL gives the similarity between a pair of embeddings of same sample obtained by two separate embedding extractors. These embedding extractors are designed to capture similar information about the channel, but dissimilar LID-specific information in the speech. Furthermore, the proposed WSSL improves the noise-robustness of the LID-network by suppressing the background noise in the speech to some extent. We demonstrate the effectiveness of the proposed approach in both seen and unseen channel conditions using a set of datasets having significant channel-mismatch.

***Index Terms*—** channel-mismatch, spoken language identification, within-sample similarity loss, domain-mismatch

1. INTRODUCTION

State-of-the-art spoken language identification (LID) networks are vulnerable to channel-mismatch occurring due to the mismatch in the channels (like recording devices, type of encoding, etc.,) used to collect the training and testing samples. Though the impact of channel-mismatch can be reduced by having sufficient training samples with different channel conditions, collecting such kind of a training dataset is a difficult task. Hence, we need to utilize the available training dataset in a best possible way so that the robustness of the LID-network can be improved. Encouraging the LID-network to learn a channel-invariant representation of the speech using adversarial multi-task learning (AMTL) is one such way.

Recently, AMTL has been used successfully to enforce domain-invariance in many applications like speech recognition [1], speaker verification/recognition [2, 3, 4, 5], speech emotion recognition [6], etc. In AMTL approach, the overall network is typically considered to have three parts: a front-end feature extractor, a primary classifier for the main task and an auxiliary domain classifier for identifying the domain of the training samples [7]. By using training samples belonging to multiple domains with corresponding domain labels, AMTL forces the network to learn features that are discriminative for the main task but invariant across domains. For example, in [8, 9], the domain classifier is trained with labels “source” and “target” to obtain features that are invariant across these two domains. Similarly, in [2], domain classifier is trained with labels like “Quiet”, “TV”, and “Music” to obtain features invariant across different types of background conditions. The adversarial training between the feature extractor and the domain classifier in AMTL forces the features to be invariant across the given domains.

However, AMTL approach cannot be used when the training samples do not contain the corresponding domain labels. For example, if a LID dataset does not contain the corresponding channel/speaker labels for the training samples, then AMTL cannot used to learn a channel/speaker-invariant representation. Furthermore, some LID datasets do not contain any channel diversity, as all samples are obtained from same type of channel (recording device). For example, dataset in [10] contains speech samples collected only through telephone lines, [11, 12] contains samples collected using mobile phones only and [13] contains samples recorded only using high quality recording devices in a controlled environment. Again, AMTL cannot be used in this case to learn channel-invariance, as it requires samples from at least two separate domains to enforce invariance.

Motivated by these, we propose a novel within-sample similarity loss (WSSL), which encourages the LID-network to learn a channel-invariant representation of the speech without using any channel labels. Specifically, the WSSL gives the similarity between a pair of utterance-level embeddings of same speech sample that are obtained using two separate embedding extractors. These embedding extractors are designed to capture similar information about the channel-specific contents in the speech (that remain almost constant throughout the utterance) and dissimilar information about the LID-specific contents in the speech. Using WSSL, the network is encouraged to suppress the similarities between these two embeddings, leading to suppression of channel related contents in the speech. Furthermore, the proposed WSSL encourages the two embedding extractors in the network to suppress the background noise to some extent and encode complementary LID-specific contents in the speech, leading to better performance even in noisy background conditions.

We demonstrate the effectiveness of the proposed approach using a set of training and testing datasets having significant differences in their characteristics. We also release both training and testing datasets used in this paper so that the obtained results can be reproduced.

2. PROPOSED FRAMEWORK

The block diagram of the proposed approach is given in Fig.1. It contains a feature extractor block to extract an utterance-level embedding (represented as u-vector in Fig.1) of the input speech with parameters (weights and biases of the network) $\boldsymbol{\theta_F}$ and a language classifier block with parameters $\boldsymbol{\theta_C}$. During training, the parameters of the LID-network $\boldsymbol{\theta_{net}} = \{\boldsymbol{\theta_F}, \boldsymbol{\theta_C}\}$ are tuned such that the ob-

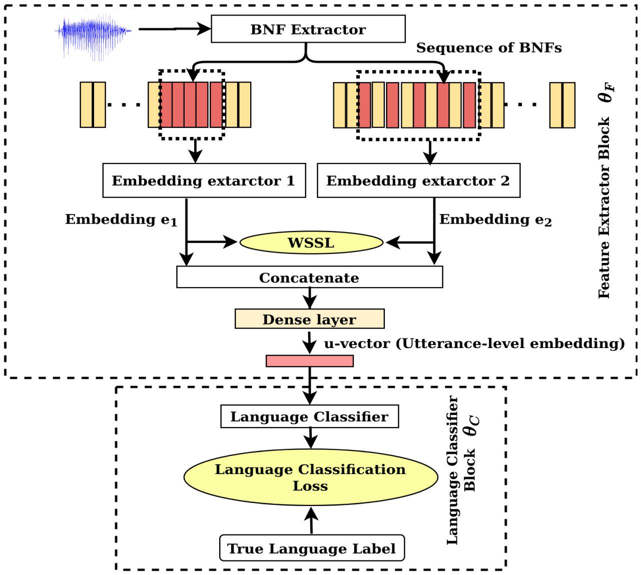

Fig. 1: Block diagram of the proposed approach with WSSL. Orange coloured frames in sequence of BNFs indicate the frames selected as input within an analysis window.

tained u-vector carries only LID-specific contents in the speech by ignoring channel-specific contents.

2.1. Feature extractor block for obtaining fixed-length u-vector

The role of the feature extractor block ($\boldsymbol{\theta_F}$) is to provide an utterance-level embedding (represented as u-vector in Fig. 1) of the input speech sample. This block contains a pre-trained bottleneck feature (BNF) extractor at the front-end to convert the speech utterance into a sequence of 80-dimensional BNFs [14]. This BNF extractor was originally trained for identifying 3096 phone states from 17 languages. Each of the extracted BNFs covers a total context of 31 frames (325 ms) of input speech [14]. Since BNFs are extracted for every input speech frame, successive BNFs have an overlap of 30 frames (96% overlap).

The input sequence of BNFs is then analyzed by two embedding extractors to provide two utterance-level embeddings (represented as $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$ in Fig. 1) of the speech. The architecture of the embedding extractor is shown in Fig. 2, which is motivated by the network in [15] and [16]. It contains two bidirectional long short-term memory (BLSTM) layers with 256 and 64 nodes respectively in first and second layer. These BLSTM layers analyze the input sequence of BNFs by dividing it into fixed-length chunks (with 50% overlap between successive chunks) to generate LID-seq-senones [15]. These LID-seq-senones are nothing but the activation obtained at the output of second BLSTM layer for each chunk of BNF vectors [15]. The mean and standard deviation of these LID-seq-senones are then computed using a statistics pooling layer. The concatenated mean and standard deviation are then passed through a dense layer with 128 nodes to get the utterance-level embedding of the speech (represented as $\boldsymbol{e_1}$ or $\boldsymbol{e_2}$ in Fig. 1).

The only difference between these two embedding-extractors is in the way they analyze the input BNF sequence. While the embedding extractor-1 analyzes the sequence of BNFs by dividing them into fixed-length chunks of 0.5 seconds, the embedding extractor-2 uses chunks of 1 second. Further, embedding extractor-1 considers all BNF vectors within the chunk as input (high resolution within small context of 0.5 seconds) whereas embedding extractor-2 uses only alternative BNF vectors within the 1 second chunk (low resolution within context of 1 second). Since successive BNFs have high overlap, the amount of information loss due to this skipping is very less; though they encode information differently.

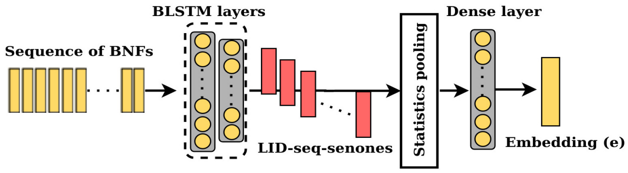

Fig. 2: Block diagram of the embedding extractor used for obtaining the embedding $\boldsymbol{e_1}$ or $\boldsymbol{e_2}$.

The motivation for analyzing the input at two different resolutions is the following. It can be observed that, in a given clean speech sample, the channel related contents remain constant throughout the utterance. Hence, the input sample can be assumed as a combination of a fast changing component (the foreground speech) and a constant (or slowly changing) component related to channel. Since two embedding extractors analyze the input at different resolutions, they capture dissimilar information about the foreground speech and similar information about the channel which remains constant. This variation between $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$ allows us to use the proposed WSSL, which we will explain in the section 2.1.1.

The outputs from two embedding extractors are then concatenated and passed through a dense layer. Output of this dense layer gives a compact utterance-level representation of the input speech sample (represented as u-vector in Fig.1). Note that, this u-vector can be directly fed to a language classifier to form an end-to-end LID-network as in Fig. 1 (excluding WSSL block). Let N be the number of nodes in the output layer of this classifier with *softmax* activation. For an input sample $\boldsymbol{x}$, let the output of the language classifier be $P(y_l|\boldsymbol{x}, \boldsymbol{\theta_F}, \boldsymbol{\theta_C})$; where $y_l \in \{l_1, l_2, ..., l_N\}$ denote the language classes. For the training sample $\boldsymbol{x}$, we can compute the cross entropy loss for the language classifier as:

$$L_l(\boldsymbol{\theta_F}, \boldsymbol{\theta_C}) = -log(P(y_l|\boldsymbol{x}, \boldsymbol{\theta_F}, \boldsymbol{\theta_C})). \qquad (1)$$

Note that, training the LID-network with only language classification loss (Eq.1) does not prevent the network from encoding the channel-specific contents in the speech. This effect is more significant when the training dataset contains very limited channel diversity. To improve the robustness of the LID-network towards an unseen target domain, the network should be trained to learn a channel-independent representation of the speech. We propose a novel within-sample similarity loss for achieving this.

2.1.1. Proposed within-sample similarity loss (WSSL)

In order to encourage the feature extractor block to suppress the channel-specific contents in the input speech, we propose an auxiliary loss function called within-sample similarity loss (WSSL). The WSSL is used along with the primary language classification loss (Eq. 1). Since the channel related contents in the speech do not change within a given speech utterance, the two embedding extractors capture similar information about this. By penalizing the network for capturing similar information in the two embeddings $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$, the network learns to suppress the contents that are common to both embeddings. Simultaneously, the primary language classification loss continuously forces the network to capture more and more LID-specific contents. Due to the combined action of these two losses, the network learns to capture complementary (dissimilar) LID-specific information in two embeddings and ignore other contents that are common to both the embeddings. Note that, unlike AMTL, WSSL does not require any channel labels for suppressing channel information. We define the WSSL as follows:

$$L_w(\boldsymbol{\theta_F}) = \alpha L_{cos}(\boldsymbol{e_1}, \boldsymbol{e_2}) - \beta L_2(\boldsymbol{e_1}, \boldsymbol{e_2}) \tag{2}$$

Where, $L_{cos}(\boldsymbol{e_1}, \boldsymbol{e_2}) = \frac{\boldsymbol{e_1} \cdot \boldsymbol{e_2}}{\|\boldsymbol{e_1}\| \|\boldsymbol{e_2}\|}$ and $L_2(\boldsymbol{e_1}, \boldsymbol{e_2}) = \|\boldsymbol{e_1} - \boldsymbol{e_2}\|_2$ are respectively the cosine similarity and Euclidean distance between the embeddings. The scalar values α and β are the trade-off parameters used to decide the impact of $L_{cos}(\boldsymbol{e_1}, \boldsymbol{e_2})$ and $L_2(\boldsymbol{e_1}, \boldsymbol{e_2})$ respectively in the training process.

2.2. The training procedure

We train the LID-network with the following total loss function.

$$L_T(\boldsymbol{\theta_F}, \boldsymbol{\theta_C}) = L_l(\boldsymbol{\theta_F}, \boldsymbol{\theta_C}) + L_w(\boldsymbol{\theta_F}) \tag{3}$$

During the training, the parameters of the LID-network $\boldsymbol{\theta_{net}} = \{\boldsymbol{\theta_F}, \boldsymbol{\theta_C}\}$ are optimized such that the network encodes complementary LID-specific contents in the speech and ignores all irrelevant contents in the input that are common to both the embeddings. This improves the robustness of the LID-network.

3. DATASETS USED IN THE STUDY

In this study, we consider a set of closely related 8 Indian languages namely, Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Odia and Telugu. To highlight the effect of channel-mismatch, we use a set of datasets having significant differences in their characteristics. First one is the IIT-Mandi read speech dataset which contains clean audio files from news broadcasts. These audio samples are recorded in a controlled environment using high quality recording devices. It contains about 38 hours of speech with approximately 4.5 hours of speech in each language. There are around 15 speakers in each language. In each language, we use approximately 4 hours of speech for training (*Readsp-Train*) and remaining portion for testing (*Readsp-Test*). Since the background conditions in the *Readsp-Test* are same as that in train set, this is called **seen** test set.

Since the amount of speech and number of speakers in *Readsp-Train* are limited, we additionally use a part of the IIIT-Hyderabad dataset [13, 17] for training in some of our experiments. Like *Readsp-Train*, this dataset also contains read speech samples recorded in controlled environment. This dataset is available upon request. In each language, we use approximately 6 hours of speech (about 25 speakers in each language) from this dataset. By combining samples from IIIT-Hyderabad dataset, we get a combined training dataset called *combined-Readsp-Train*, having approximately 10 hours of speech in each language with at least 40 speakers.

Another dataset is the IIT-Mandi YouTube dataset (*YouTube-Test*) which contains spontaneous/teaching style speech from personal interviews and educational videos obtained from YouTube. These samples are collected using different types of recording devices with different real-world background conditions. Hence, there is significant channel-mismatch between the training dataset (*Readsp-Train*) and *YouTube-Test* datasets. In each language, it contains about 1 hour of speech from at least 10 speakers. In this work, this dataset is used only for testing. Since the background conditions in this dataset are unseen by the network during the training, this represents the **unseen** test set. All speech samples used in this work have a duration between 2 to 12 seconds. We release both *Readsp-Train/Test* as well as *YouTube-Test* datasets with this paper[1].

4. RESULTS AND DISCUSSION

In this section, we study the effectiveness of the proposed method. Performances are given in accuracy (%) and the C_{avg} (%) metric [18]. Lower values of C_{avg} indicates better performance.

Here, we first demonstrate the effectiveness of the proposed WSSL in improving the channel-invariance of the LID-network by training the network using clean speech. Followed by this, the effectiveness of the WSSL in improving the noise-robustness is shown using samples with artificially added noise.

4.1. Training with clean speech samples

In this section, we train our networks using only clean speech samples and test their performance on samples from *Readsp-Test* and samples from *YouTube-Test*.

4.1.1. *Baseline systems*

To compare the effectiveness of the proposed WSSL, we use 2 baselines that are trained using only language classification loss (without WSSL). First one is the x-vector based LID system [19]. It processes the sequence of BNFs to obtain 512-dimensional x-vector which are then classified using a Gaussian back-end [19]. The clean speech samples from *Readsp-Train* dataset (having approximately 4 hours of speech per language) is used for training this system. Results obtained for this system on **seen** and **unseen** test sets are shown in 1^{st} row of Table 1.

Table 1: Performance in accuracy (*Acc*) and C_{avg} of baselines and LID-systems with WSSL in seen and unseen test sets.

LID system	seen test set *Readsp-Test*		unseen test set *YouTube-Test*	
	Acc	C_{avg}	*Acc*	C_{avg}
x-vector	83.25	9.05	62.50	22.50
Lnet_baseline	84.50	8.60	62.85	22.25
Lnet_baseline_comb	87.45	7.00	65.45	19.15
Lnet_WSSL	88.90	6.45	68.10	17.90
Lnet_WSSL_comb	90.04	6.05	71.90	15.25
Lnet_fine_tuned	81.20	11.20	79.55	10.10

Second one is a baseline LID-network (Lnet_baseline) which contains only feature extractor and language classifier blocks in Fig.1 (excluding WSSL block). The language classifier contains a dense layer with 128 nodes with *tanh* activation followed by the output layer. Results obtained for this system when trained on *Readsp-Train* are given in 2^{nd} row of Table 1. For the comparison, we also trained a baseline LID-network using the *combined-Readsp-Train* dataset (containing 10 hours of speech per language with more speakers). Results obtained for this network (Lnet_baseline_comb) are given in the 3^{rd} row. In this case, the number of nodes in each hidden layer in the network has been increased to twice as the training dataset contains more diversity.

It is seen that, both x-vector and Lnet_baselines perform very well on clean samples from *Readsp-Test* but, poorly on *YouTube-Test*. Since both x-vector and Lnet_baseline systems have seen only clean speech samples with limited channel diversity during the training, both of them have become vulnerable to unseen channel conditions in the *YouTube-Test*. In spite of using a combined training dataset with more speaker diversity, the Lnet_baseline_comb has provided only slight improvement in performance in unseen test set. This clearly indicates that, simple data-augmentation alone is not

[1] https://speechiitmandi.github.io/air/

sufficient; we need a sophisticated training strategy too, to improve channel-invariance of the LID-network.

4.1.2. Effectiveness of the proposed WSSL

Here, we experiment by including the WSSL in the training process as in Eq. 3. The value of trade-off parameters α and β (in Eq. 2) are empirically set as 0.35 and 0.15 respectively. The 4^{th} row in Table 1 shows the performance of the network trained on *Readsp-Train* (Lnet+WSSL). The performance of network trained on *combined-Readsp-Train* (Lnet+WSSL_comb) is given in 5^{th} row.

It is seen that, both Lnet+WSSL_comb and Lnet+WSSL have performed significantly better than baselines on unseen test set. Clearly, the inclusion of WSSL has allowed the networks to learn a channel-invariant representation, leading to significantly better performance. Note that, both these systems have performed better in seen test set too, compared to baseline systems. This clearly indicates that, WSSL encourages the network to capture complementary LID-specific contents in two embeddings and it does not suppress any LID-specific contents in the speech.

For the purpose of comparison, we have also given the performance of a LID-network that has been fine-tuned to the *YouTube-Test* dataset. In this, we have used approximately 30% samples from *YouTube-Test* for fine-tuning a pretrained network (Lnet_baseline). During the fine-tuning process, the parameters of only last two layers of the Lnet_baseline has been modified, keeping all other layers unchanged. The results obtained for this system (Lnet_fine_tuned) when tested on the remaining 70% samples from *YouTube-Test* as well as on *Readsp-Test* are given in 6^{th} row of Table 1.

Note that, unlike Lnet_fine_tuned system, our proposed WSSL approach does not use any samples from the *YouTube-Test* for training/fine-tuning. The comparison with Lnet_fine_tuned is done only for showing the effectiveness of the proposed WSSL. In spite of not using any samples from *YouTube-Test* for training/fine-tuning, the performance of Lnet_WSSL is comparable to that of Lnet_fine_tuned on unseen test set.

Not surprisingly, the Lnet_fine_tuned has performed slightly poorer on seen test set compared to the performance of Lnet_baseline. This is because, the parameters of Lnet_fine_tuned have been optimized for the *YouTube-Test* during the fine-tuning process. Hence, samples from *Readsp-Test* have slight channel-mismatch with this network. Note that, unlike Lnet_fine_tuned, the performance of Lnet_WSSL does not degrade on seen test set.

Apart from channel-mismatch, the LID networks are also affected by background noise in the speech. In the next section, we evaluate the effectiveness of WSSL in noisy conditions.

4.2. Training with noisy speech samples

Note that, the *Readsp-Train* used for training contains only clean speech samples. To simulate the speech samples collected in various indoor and outdoor scenarios, we artificially add samples from 4 acoustic scene classes from the DCASE-2017 scene classification dataset [20, 21]. We used 70% audio samples from *Lakeside beach, Bus, Car* and *City center* classes to corrupt the train data (*Readsp-Train*) and remaining 30% samples to corrupt the *Readsp-Test*. The addition of 4 types of noise results in 4 fold increase in the size of both train and test sets. We denote the resulting training and testing datasets respectively as *Rdsp-Noise-Train* and *Rdsp-Noise-Test*. Due to the addition of noise, the SNR of the resulting speech samples varies between 0.5db to 14.5db with a mean of 4.32db. Note that, we do not artificially add noise to *YouTube-Test* samples, as they already contain some real-world background noise.

Table 2: Performance of LID-networks with WSSL and AMTL in seen and unseen test sets when trained using noisy speech samples.

LID system	seen test set *Rdsp-Noise-Test*		unseen test set *YouTube-Test*	
	Acc	C_{avg}	Acc	C_{avg}
Lnet_baseline_noisy	65.60	20.25	63.40	21.60
Lnet_WSSL_noisy	68.80	18.20	69.15	16.85
Lnet_AMTL_noisy	73.50	14.15	64.50	19.90
LnetWSSL_AMTL_noisy	75.70	13.60	70.35	15.95

As the *Rdsp-Noise-Train* contains more complexity than *Readsp-Train*, all networks trained on this dataset contain thrice the number of nodes in each hidden later compared to their counterparts trained on *Readsp-Train*. The 1^{st} row in Table 2 shows the results obtained for the baseline network Lnet_baseline_noisy (trained on *Rdsp-Noise-Train*), on *Rdsp-Noise-Test* and *YouTube-Test* sets. The results obtained for the network trained using WSSL (Lnet_WSSL_noisy) are also given in 2^{nd} row. The Lnet_WSSL_noisy has performed significantly better than Lnet_baseline_noisy and slightly better than Lnet_WSSL (given in Table 1) in unseen test set. It has provided about 3% improvement in accuracy in seen test set too, compared to the Lnet_baseline_noisy. This indicates that, apart from channel-specific contents, the WSSL also suppresses the noise in the speech to some extent.

4.2.1. Comparison with AMTL approach

Since the noise samples are added artificially, the training samples in *Rdsp-Noise-Train* contain the corresponding noise identity. This allows us to build a noise-invariant LID-network using AMTL approach [1, 2]. In this case, the LID-network contains an additional domain classifier for identifying the noise type in the speech. Like language classifier, the domain classifier also takes the u-vector as input. The domain classifier contains a dense layer with 256 nodes with *tanh* activation followed by output layer with 5 nodes to represent the background conditions in the speech (clean speech and 4 types of noise). The results obtained for the network trained using AMTL (Lnet_AMTL_noisy) are given on the 3^{rd} row of Table 2.

It is seen that, the Lnet_AMTL_noisy has performed significantly better on *Rdsp-Noise-Test* compared to Lnet_WSSL_noisy. Unlike channel information, the background noise is mostly non-stationary in nature. Hence, two embedding extractors capture dissimilar information about the noise, due to which, WSSL is less effective in suppressing the noise. However, on unseen test set in which channel-mismatch is more dominant than noise, the network with WSSL has given much better performance than Lnet_AMTL_noisy. The last row in Table 2 shows the performance of a network trained with both WSSL and AMTL. This system has performed well on both noisy speech as well as in channel-mismatched case. While AMTL improves the noise-invariance of the network, the WSSL improves its channel-invariance, leading to better performance in both the cases.

5. CONCLUSIONS

In this paper, we proposed a novel within-sample similarity loss (WSSL) for improving the channel-invariance of the LID-network. The results obtained indicate that the proposed WSSL significantly improves the performance in channel-mismatched conditions. Also, the combination of WSSL with AMTL significantly improves the robustness of the LID-network to unseen target domain.

6. REFERENCES

[1] Y. Shinohara, “Adversarial multi-task learning of deep neural networks for robust speech recognition.” in *INTERSPEECH-2016*, pp. 2369–2372.

[2] Z. Meng, Y. Zhao, J. Li, and Y. Gong, “Adversarial speaker verification,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-2019)*, pp. 6216–6220.

[3] G. Suthokumar, V. Sethu, K. Sriskandaraja, and E. Ambikairajah, “Adversarial multi-task learning for speaker normalization in replay detection,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-2020)*, pp. 6609–6613.

[4] H. Wang, H. Dinkel, S. Wang, Y. Qian, and K. Yu, “Cross-domain replay spoofing attack detection using domain adversarial training.” in *INTERSPEECH-2019*, pp. 2938–2942.

[5] S. Wang, J. Rohdin, L. Burget, O. Plchot, Y. Qian, K. Yu, and J. Cernockỳ, “On the usage of phonetic information for text-independent speaker embedding extraction.” in *INTERSPEECH-2019*, pp. 1148–1152.

[6] M. Abdelwahab and C. Busso, “Domain adversarial for acoustic emotion recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 12, pp. 2423–2435, 2018.

[7] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *International Conference on Machine Learning (ICML-2015)*, pp. 1180–1189.

[8] 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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-2018)*, pp. 4889–4893.

[9] G. Bhattacharya, J. Alam, and P. Kenny, “Adapting end-to-end neural speaker verification to new languages and recording conditions with adversarial training,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-2019)*, pp. 6041–6045.

[10] Y. K. Muthusamy, R. A. Cole, and B. T. Oshika, “The OGI multi-language telephone speech corpus,” in *Second International Conference on Spoken Language Processing*, 1992.

[11] D. Wang, L. Li, D. Tang, and Q. Chen, “AP16-OL7: A multilingual database for oriental languages and a language recognition baseline,” in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA-2016)*, pp. 1–5.

[12] 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,” *arXiv preprint arXiv:2006.03473*, 2020.

[13] K. Mounika, S. Achanta, H. Lakshmi, S. V. Gangashetty, and A. K. Vuppala, “An investigation of deep neural network architectures for language recognition in Indian languages.” in *INTERSPEECH-2016*, pp. 2930–2933.

[14] A. Silnova, P. Matejka, O. Glembek, O. Plchot, O. Novotny, F. Grezl, P. Schwarz, L. Burget, and J. Cernocky, “BUT/Phonexia Bottleneck Feature Extractor,” in *The Speaker and Language Recognition Workshop Odyssey-2018*, pp. 283–287.

[15] H. Muralikrishna, S. Pulkit, J. Anuksha, and A. D. Dileep, “Spoken language identification using bidirectional LSTM based LID Sequential Senones,” in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU-2019)*, pp. 320–326.

[16] A. Lozano-Diez, O. Plchot, P. Matejka, and J. Gonzalez-Rodriguez, “DNN based embeddings for language recognition,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-2018)*, pp. 5184–5188.

[17] R. K. Vuddagiri, H. K. Vydana, and A. K. Vuppala, “Curriculum learning based approach for noise robust language identification using DNN with attention,” *Expert Systems with Applications*, vol. 110, pp. 290–297, 2018.

[18] “The 2015 NIST Language Recognition Evaluation plan (LRE15),” *https://www.nist.gov/itl/iad/mig/2015-language-recognition-evaluation*.

[19] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, “Spoken language recognition using x-vectors,” in *The Speaker and Language Recognition Workshop Odyssey-2018*, pp. 105–111.

[20] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, “DCASE 2017 challenge setup: Tasks, Datasets and Baseline system,” 2017.

[21] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, “Detection and Classification of Acoustic Scenes and Events: Outcome of the DCASE 2016 challenge,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, 2018.