Convolutional Neural Networks and Language Embeddings for End-to-End Dialect Recognition

Suwon Shon[1], Ahmed Ali[2], James Glass[1]

Massachusetts Institute of Technology, Cambridge, MA, USA[1]
Qatar Computing Research Institute, HBKU, Doha, Qatar[2]

{swshon,glass}@mit.edu   amali@hbku.edu.qa

Abstract

Dialect identification (DID) is a special case of general language identification (LID), but a more challenging problem due to the linguistic similarity between dialects. In this paper, we propose an end-to-end DID system and a Siamese neural network to extract language embeddings. We use both acoustic and linguistic features for the DID task on the Arabic dialectal speech dataset: Multi-Genre Broadcast 3 (MGB-3). The end-to-end DID system was trained using three kinds of acoustic features: Mel-Frequency Cepstral Coefficients (MFCCs), log Mel-scale Filter Bank energies (FBANK) and spectrogram energies. We also investigated a dataset augmentation approach to achieve robust performance with limited data resources. Our linguistic feature research focused on learning similarities and dissimilarities between dialects using the Siamese network, so that we can reduce feature dimensionality as well as improve DID performance. The best system using a single feature set achieves 73% accuracy, while a fusion system using multiple features yields 78% on the MGB-3 dialect test set consisting of 5 dialects. The experimental results indicate that FBANK features achieve slightly better results than MFCCs. Dataset augmentation via speed perturbation appears to add significant robustness to the system. Although the Siamese network with language embeddings did not achieve as good a result as the end-to-end DID system, the two approaches had good synergy when combined together in a fused system.

1. Introduction

A significant step forward in speaker and language identification (LID) was obtained by combining i-vectors and Deep Neural Networks (DNNs) [1, 2, 3]. The task of dialect identification (DID) is relatively unexplored compared to speaker and language recognition. One of the main reasons is due to a lack of common datasets, while another reason is that DID is often regarded as a special case of LID, so researchers tend to concentrate on the more general problems of language and speaker recognition. However, DID can be an extremely challenging task since the similarities among language dialects tends to be much higher than those used in the more general LID task.

Arabic is an appropriate language for which to explore DID, due to its uniqueness and widespread use. While 22 countries in the Arab world use Modern Standard Arabic (MSA) as their official language, citizens use their own local dialect in their everyday life. Arabic dialects are historically related and share Arabic characters, however, they are not mutually comprehensible. Arabic DID therefore poses different challenges compared to other language dialects containing comprehensible vernacular. Arabic dialects are challenging to distinguish because they belong to the same language family. Arabic dialects typically share a common phonetic inventory and other linguistic feature like characters, so words and phonemes can be utilized by automatic speech recognition (ASR) contrary to general LID.

The natural language processing (NLP) community has tended to partition the Arabic language into 5 broad categories: Egyptian (EGY), Levantine (LEV), Gulf (GLF), North African (NOR), and MSA. The Multi Genre Broadcast(MGB) challenge[1] committee established an Arabic dialect dataset in 2016, and holds a DID challenge series that attracts interest in Arabic DID from researchers in both the NLP and speech communities. The MGB-3 dataset contains 5 dialects with 63.6 hours of training data and 10.1 hours of test data. A DID benchmark of the task was attained using the i-vector framework using bottleneck features [4]. Using linguistic features such as words and characters results in similar performance to that obtained via acoustic features with a Convolutional Neural Network (CNN)-based backend [5, 6, 7]. Since the linguistic feature space is different from the acoustic feature space, a fusion of the results from both feature representations has been shown to be beneficial [6, 5, 8]. A recent study achieves 70% [9] for a single system and 80% [10] for a fused system. Given that there are only 5 dialects, these accuracies are relatively low, affirming the difficulty of this DID task.

Recently, many Deep Neural Network (DNN)-based end-to-end and speaker embedding approaches have achieved impressive results for text independent speaker recognition and LID [11, 12, 13, 14]. This research indicates that these methods perform close to or slightly better than traditional i-vectors, though it has focused on improving upon i-vectors, so there is room for further detailed analyses of various task conditions.

In this work, we propose two different approaches for DID using both acoustic and linguistic features. For acoustic features, we explore an end-to-end DID model based on a CNN with global average pooling. We examine three representations based on Mel-Frequency Cepstral Coefficients (MFCCs), log Mel-scale filter banks energies (FBANK) and spectrogram energies. We employ data augmentation via speech and volume perturbation to analyze the impact of dataset size. Finally, we analyze the effectiveness of training with random utterance segmentation, as a strategy to cope with short input utterances.

For language embeddings, we adopt a Siamese neural network [15] model using a cosine similarity metric to learn a dialect embedding space based on text-based linguistic features. The network is trained to learn similarities between the same dialects and dissimilarities between different dialects. Linguistic feature are extracted from an automatic speech recognizer

---
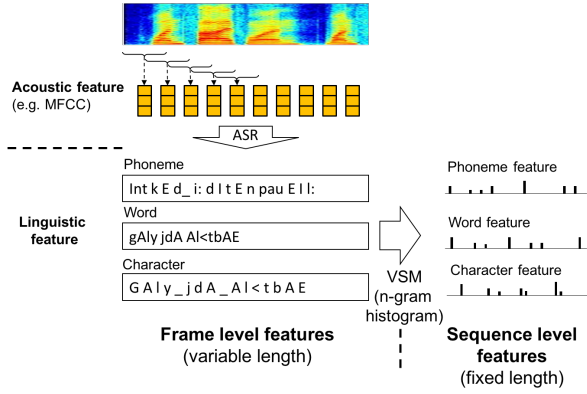
[1]www.mgb-challenge.org

10.21437/Odyssey.2018-14

Figure 1: Illustration of acoustic and linguistic feature creation.

(ASR), and based on words, characters and phonemes. Siamese neural networks are usually applied on verification tasks for end-to-end systems, however a previous study showed that this approach could be also applied on an identification task to make the original features more robust [9]. We applied this approach on linguistic features to extract language embeddings to improve DID performance while reducing the enormous feature dimensionality. Python and Tensorflow code for the end-to-end system[2] and Siamese network language embeddings[3] is available, to allow others to reproduce our results, and apply these ideas to similar LID and speaker verification tasks.

2. Dialect Identification Baseline

2.1. Acoustic features

For LID and DID, the i-vector is regarded as the state-of-the-art for general tasks, while recent studies show combinations with DNN-based approaches produce competitive performance [2]. Apart from the basic i-vector approach, bottleneck (BN) features extracted from an ASR acoustic model were successfully applied to LID [1, 16, 2, 5, 6]. Although the acoustic model is usually trained on single language, it performs reasonably well on LID tasks [16]. The mismatch does appear to matter on speaker recognition tasks however [17, 18]. A stacked BN feature can be extracted from a second DNN that has a BN layer, and its input is the output of the first BN layer with context expansion [1, 16]. A Gaussian Mixture Model - Universal Background Model (GMM-UBM) and a total variability matrix are trained in an unsupervised manner with BN features from a large dataset. Although the main purpose of LID is to obtain phonetic characteristics of each languages while suppressing individual speaker's voice characteristics, the training scheme is exactly the same as training a GMM-UBM and total variability matrix for speaker recognition. The only difference is that the subsequent projection of the i-vector is based on measuring the distances with other languages as opposed to other speakers.

2.2. Linguistic features

While the inherent similarity between dialects makes DID a more difficult task than LID, DID can take advantage of linguistic features because dialects tend to share a common phonetic inventory. Word and character sequences can be extracted using

a state-of-the-art ASR system, such as the Time delayed Neural Network (TDNN) [19] or Bidirectional Long Short-Term Memory Recurrent Neural Network (BLSTM-RNN) acoustic model with RNN-based language model rescoring. Extracted text sequence can then be converted into a Vector Space Model(VSM) [6]. The VSM represents each utterance as a fixed length, sequence level, high-dimensional sparse vector $\vec{u}$:

$$\vec{u} = (f(u, x_1), f(u, x_2), ..., f(u, x_D)) \qquad (1)$$

where $f(u, x_i)$ is the number of times word x_i occurs in utterance u, and D is the dictionary size. The vector $\vec{u}$ can also consist of an n-gram histogram. Figure 1 illustrates the linguistic feature extraction concept based on acoustic features.

For the MGB-3 Arabic dataset, the tri-gram word dictionary size D is 466k, and most of $f(u, x_i)$ is 0. Due to this high dimensionality, the kernel trick method used with Support Vector Machine's (SVMs) is a suitable approach for classification. Another linguistic feature, the character and phoneme feature, can also be extracted by an ASR system or standalone phonetic recognizer. Although both ASR and phonetic recognizers start from an acoustic input, the linguistic feature space consists of higher level features that contain semantic information of variable duration speech segments. While an ASR system, and an acoustic feature-based speaker recognition or LID/DID system are vulnerable to domain mismatched conditions [20, 21, 22, 23], the affected results in the higher-level linguistic spaces will be different. Thus, linguistic feature-based approaches could complement acoustic feature-based approaches to produce a superior fusion result.

3. End-to-End DID with Acoustic Features

Recently, end-to-end approaches which do not use i-vector techniques showed impressive results on both LID and speaker recognition [11, 12, 13, 14]. They did not start from raw waveforms, but from acoustic features ranging from MFCCs to spectrograms. The deep learning models consisted of CNNs, RNNs and Fully Connected (FC) DNN layers. Recent studies [14, 13] report combinations of CNN and FC with a global pooling layer obtains the best result for a speaker representation from text-independent and variable length speech inputs. The global pooling layer has a simple function such as averaging sequential inputs, but it effectively convert frame level representations to utterance level representations, which is advantageous for speaker recognition and LID. The size of the final softmax layer is determined by the task specific speaker or language labels. For identification tasks, the Softmax output can be used directly as a score for each class. For verification tasks, usually the activation of the last hidden layer is extracted as a speaker or language representation vector, and is used to calculate a distance between enrollment and test data.

Our end-to-end system is based on [14, 13], but instead of a VGG [24] or Time Delayed Neural Network, we used four 1-dimensional CNN (1d-CNN) layers (40×5 - 500×7 - 500×1 - 500×1 filter sizes with 1-2-1-1 strides and the number of filters is 500-500-500-3000) and two FC layers (1500-600) that are connected with a Global average pooling layer which averages the CNN outputs to produce a fixed output size of 3000×1. After global average pooling, the fixed length output is fed into two FC layers and a Softmax layer. The structure of end-to-end dialect identification system is represented in Figure 2. The four CNN layers span a total of 11 frames, with strides of 2 frames over the input. For Arabic DID, the Softmax layer size is 5. We

---

[2]https://github.com/swshon/dialectID_e2e

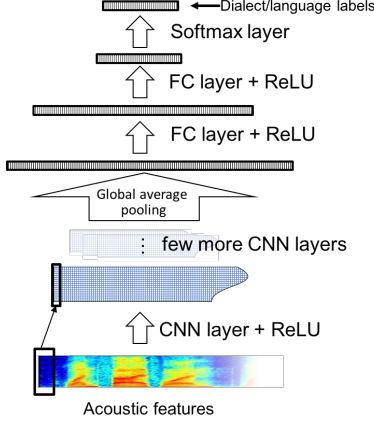[3]https://github.com/swshon/dialectID_siam

Figure 2: End-to-End dialect identification system structure.

Figure 3: Siamese neural network structure.

examined the FBANK, MFCC and spectrogram as acoustic features. Filter size on first CNN layer changes to 200×5 when we use Spectrogram. The Softmax output was used to compute similarity scores of each dialect for identification.

3.1. Internal Dataset augmentation

Neural network-based deep learning models are but the most recent example of the age-old ASR adage that "there's no data like more data." These models are able to absorb large quantities of training data, and generally do better, the more data they are exposed to. Given a fixed training data size, researchers have found that augmenting the corpus size by artificial means can also be effective. In this paper, we explored internal dataset augmentation methods. One augmentation approach is segmentation of the training dataset into small chunks [13, 14, 11]. We did not segment the dataset in advance, but did random segmentations on mini-batches. The random length was selected to have a uniform distribution between 2 and 10 seconds with 1 second intervals including original length. From previous studies, it was found that random segmentation yields better performance on short utterances [13, 14, 11]. However, the investigations did not show the performance before applying this method, so it is difficult to judge how much gains were achieved from this approach. We analyzed performances before and after applying random segmentation on various test set lengths.

The other technique for augmentation is perturbation of the original dataset in term of speed [25] which is used ASR. The idea is to perturb the audio to play slightly faster or slower. We investigated the effectiveness of speed and volume perturbation on the DID task using speed factors of 0.9 and 1.1 and volume factors of 0.25 and 2.0.

4. DID Linguistic Feature Embeddings

In previous study [5], they explored CNN back-end for the linguistic feature using cross-entropy objective function with softmax output of the network. For identification task, it is natural to consider that the cross-entropy objective function between class labels and output of the network. However, our recent study [9] shows that using the Euclidean distance loss function between the label and the cosine distance of the NN output pair which is usually adopted on a binary classification task is still useful for identification task by learning similarity and dissimilarity
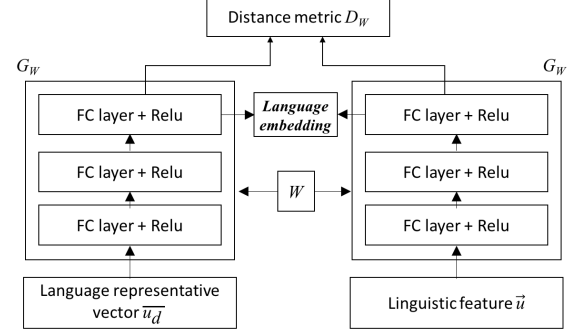
between classes. Thus, we adopted a Siamese neural network approach to extract language embeddings from linguistic features. The Siamese neural network has two parallel neural networks, G_W, which shares the same set of weights and biases W as shown in Figure 3. While the network has a symmetric structure, the input pair to the network consists of language representative vectors $\overline{u_d}$ and linguistic feature $\vec{v}$, where d is language/dialect index and i is utterance index. The language representative vector can be represented as $\overline{u_d} = (1/n_d) \sum_{i=1}^{n_d} \vec{u}_i^d$, where n_d is the number of utterances for each dialect d and $\vec{u}_i^d$ is a segment level feature of dialect d. Thus, the pair of network inputs is the combination of the language representative vectors and all linguistic features. Let Y be the label for the pair, where $Y = 1$ if the pair belongs to the same language/dialect, and $Y = -1$ if a different language/dialect. To optimize the network, we use a Euclidean distance loss function with stochastic gradient descent (SGD) between the label and the cosine similarity of the pair, D_W, where

$$L(\overline{u_d}, \vec{u}, Y) = ||Y - D_W(\overline{u_d}, \vec{u})||_2^2 \qquad (2)$$

FC layers have 1500-600-200 neurons. All layers use Rectified Linear Unit (ReLU) activations. From the network G_W, language embeddings can be extracted from both language representative vectors and linguistic features by obtaining activations from the last hidden layer (200 dimensions).

5. Dialectal Speech Dataset

The MGB-3 dataset partitions are shown in Table 1. Each partition consists of fives Arabic dialects : EGY, LEV, GLF, NOR, MSA. Detailed corpus statistics are found in [4]. Although the development set is relatively small compared to the training set, it matches the test set channel conditions, and thus provides valuable information about the test domain.

Dataset	Training	Development	Test
Utterances	13,825	1,524	1,492
Size	53.6 hrs	10 hrs	10.1 hrs
Channel (recording)	Carried out at 16kHz	Downloaded directly from a high-quality video server	

Table 1: MGB-3 Dialectal Arabic Speech Dataset Properties.

Feature	Maximum			Converged		
	Accuracy(%)	EER(%)	C_{avg}	Accuracy(%)	EER(%)	C_{avg}
MFCC	65.55	20.24	19.92	61.33	21.95	21.53
FBANK	64.81	20.22	19.91	61.26	22.12	21.79
Spectrogram	57.57	24.48	24.49	54.22	25.90	25.09

Table 2: Performance on end-to-end dialect identification system by features
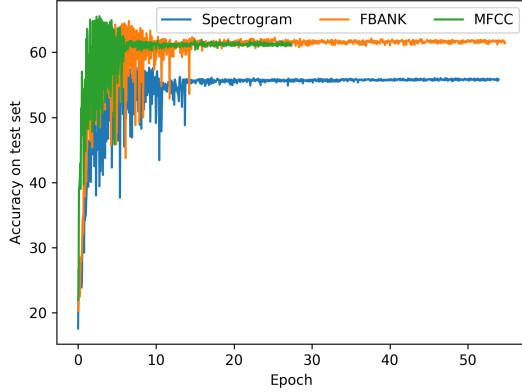

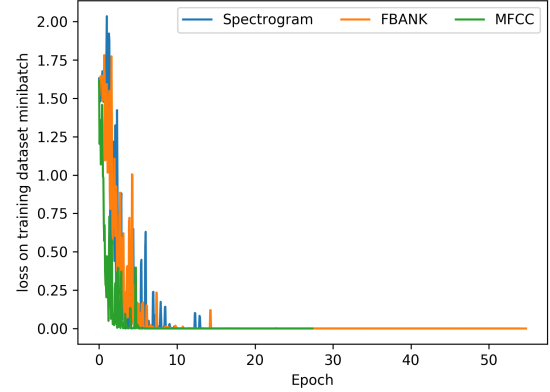Figure 4: End-to-end DID accuracy by epoch.

Figure 5: End-to-end DID mini-batch training loss by epoch.

6. DID Experiments

6.1. Implementation and Training

For i-vector implementation, the BN features were extracted from an ASR system trained for MSA speech recognition [26] with a similar configuration as [1]. A DNN that extracted BN features was trained on 60 hours of manually transcribed Al-Jazeera MSA news recordings [27]. Then, the GMM-UBM with 2,048 mixtures and a 400 dimensional i-vector extractor was trained based on the BN features. The detailed configuration is described in [6]. After extracting i-vectors, a whitening transformation and length normalization are applied. Similarity scores were calculated by Cosine Similarity (CS) and Gaussian Backend (GB). Linear Discriminant Analysis (LDA) was applied after extracting i-vectors with language labels. On this experiment, LDA reduces the 400 dimensional i-vector to 4 dimensions.

For phoneme features, we used four different phoneme recognizers: Czech (CZ), Hungarian (HU) and Russian (RU) using narrow-band models, and English (EN) using a broadband model [28]. For words and character features, word sequences are extracted using an Arabic ASR system built as part of the MGB-2 challenge on Arabic broadcast speech [29]. From the word sequence, character sequences can also be obtained by splitting words into characters. Finally, phoneme, character and word feature can be represented in VSM by using each sequence. We use unigrams for word features, and trigrams for character and phoneme feature. The word and character feature dimensions are 41,657 and 13,285, respectively. Phoneme feature dimensions are 47,466 (CZ), 50,320 (HU), 51,102 (RU) and 33,660 (EN). An SVM was used to measure similarity between the test utterance and 5 dialects [6].

To train the end-to-end dialect identification system, we used 3 features: MFCC, FBANK and spectrogram. The structure of the DNN is shown in Figure 2 and described fully in Section 3. The SGD learning rate was 0.001 with decay in every 50000 mini-batches with a factor of 0.98. ReLUs were used for activation nonlinearities. We used both the training and development dataset for training the DNN and excluded 10% of development dataset of each dialect for a validation set. For acoustic input, the FFT window length was 400 samples with a 160 sample hop which is equivalent to 25ms and 10ms respectively for 16kHz audio. A total of 40 coefficients were extracted for MFCCs and FBANKs, and 200 for spectrograms. All features were normalized to have zero-mean and unit variance.

To train the Siamese network, we made utterance pairs from the training and development sets. Since the number of true ("same dialect") pairs is naturally lower than false ("different dialect") pairs, we adjust the training batch to have the same 50%:50% ratio of true and false pairs. Since the test set domain is mismatched with the training set, the development dataset is very important because it contains channel information of the target domain. Thus, we expose development set pairs more than training set pairs by a factor of 5.

All scores were normalized by Z-norm for each dialect, with fusion was done by logistic regression. Performance was measured in accuracy, Equal Error Rate (EER) and minimum decision cost function C_{avg}. Accuracy was measured by taking the dialect showing the maximum score between a test utterance and the 5 dialects. Minimum C_{avg} was computed from hard decision errors and a fixed set of costs and priors from [30]

6.2. Experimental Result and Discussion

For acoustic features, Figures 4 and 5 show the test set accuracy and mini-batch loss on training set for corresponding epochs. From the figures, it appears that the test set accuracy is not

Augmentation method	Maximum			Converged		
(feature = MFCC)	Accuracy(%)	EER(%)	C_{avg}	Accuracy(%)	EER(%)	C_{avg}
Volume	67.49	20.37	20.00	62.47	21.55	21.08
Speed	70.51	17.54	17.39	65.42	19.87	19.19
Volume and speed	70.91	17.79	17.93	67.02	19.37	19.01

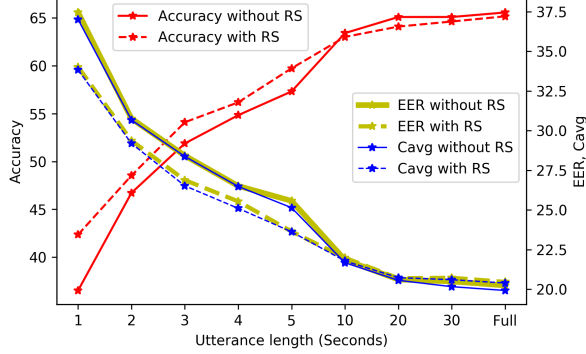Table 3: Performance on end-to-end DID system using augmented dataset.

Figure 6: Performance comparison with and without Random Segmentation (RS) on training dataset.
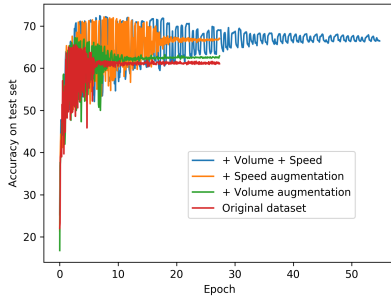

Figure 7: DID accuracy by epoch using augmented data.

Feature (on augmented dataset)	Accuracy(%)	EER(%)	C_{avg}
MFCC	70.91	17.79	17.93
FBANK	71.92	18.01	17.63
Spectrogram	68.83	18.70	18.69

Table 4: Performance on end-to-end DID system using augmented dataset.

utterance is very short such as under 10 seconds. For over 10 seconds or original length, it still shows performance improvement because random segmentation provides diversity given a limited training dataset. However the impact is not as great as on short utterances.

Figure 7 and Table 3 show the performance on data augmented by speed and volume perturbation. The accuracy fluctuated similarly to that observed on the original data as in Figure 4, but all numbers are higher. Volume perturbation shows slightly better results than using the original dataset in both the Maximum and Converged condition. However, speed perturbation shows impressive gains on performance, particularly, a 15% improvement on EER. Using both volume and speed seem to have advantages when the network has converged, but it need more epochs to converge because of the larger data size compared to volume or speed perturbation.

Table 4 shows a performance comparison on different features when using volume and speed data augmentation. The performance gains from dataset augmentation are 8%, 11% and 20% in accuracy for MFCC, FBANK and spectrogram respectively. Spectrograms still attain the worst performance, as on the original dataset (table 2). However it is somewhat surprising that the gain from increasing the dataset size is much higher than for MFCCs which achieve a relatively small increase. Table 5 shows performance comparisons between i-vectors and the end-to-end approach. A combination of i-vectors and LDA shows better performance on EER and C_{avg} than the original i-vector. However, the end-to-end system in Table 2 (which uses the same original dataset as the i-vector system) outperforms the i-vector-LDA for all features.

All end-to-end systems in Table 5 made use of random seg-

exactly correlated with loss and all three features show better performance before the network converged. We measured performance for two conditions denoted as "Maximum" and "Converged" in Table 2. The maximum condition means the network achieves the best accuracy on the validation set. The converged condition means the average loss of 100 mini-batches goes under 0.00001. From the table, the difference between maximum and converged is 5-10% in all scenarios, so a validation set is essential to judge the performance of the network and should always be monitored to stop the training.

MFCC and FBANK features achieve similar DID performance while the spectrogram is worse. Theoretically, spectrograms have more information than MFCCs or FBANKs, but it seems hard to optimize the network using the limited dataset. To obtain better performance on spectrograms, it seems to require using a pre-trained network on an external large dataset like VGG as in [13].

Figure 6 shows performance gains from random segmentation. Random segmentation shows most gain when the test

System	Accuracy(%)	EER(%)	C_{avg}
i-vector (CDS)	60.32	26.98	26.35
i-vector (GB)	58.58	22.50	22.56
i-vector-LDA (CDS)	62.60	21.05	20.12
i-vector-LDA (GB)	63.94	19.45	19.17
End-to-End (MFCC)	71.05	18.01	17.97
End-to-End (FBANK)	**73.39**	**16.30**	**15.96**
End-to-End (Spectrogram)	70.17	17.64	17.27

Table 5: Acoustic feature performance measurement on Arabic dialect MGB-3 dataset. All end-to-end systems were trained using random segmentation and volume/speed data augmentation.

Phoneme Recognizer	System	Accuracy(%)	EER(%)	C_{avg}
Hungarian	Baseline	48.86	29.94	29.16
	Embedding	**54.49**	**28.69**	**27.77**
Russian	Baseline	45.04	31.30	30.65
	Embedding	44.24	38.09	36.69
English	Baseline	33.78	39.46	39.04
	Embedding	48.86	42.02	36.76
Czech	Baseline	45.64	31.85	31.16
	Embedding	53.55	29.26	28.67

Table 6: Phoneme feature based on 4 different phoneme recognizer performance measurement.

Feature	System	Accuracy(%)	EER(%)	C_{avg}
Character	Baseline	51.34	30.03	30.17
	Embedding	**58.18**	**25.48**	**25.68**
Word	Baseline	50.00	30.73	30.41
	Embedding	**58.51**	**24.87**	**24.99**

Table 7: Character and word feature performance measurement.

mentation and volume/speed data augmentation during training. Since the internal dataset augmentation approach gives diversity on limited datasets, the end-to-end system could be significantly improved without any additional dataset. It is interesting that the most refined feature, MFCC, shows less performance improvement after dataset augmentation was applied (8% in accuracy) than less refined, more raw features, i.e. FBANK and spectrogram show comparatively higher improvement by 14% and 22%. Finally the spectrogram feature performs slightly better than MFCC in EER and C_{avg}. It implies that if we have large dataset, we can use raw signals as input features. At the same time, however, it is difficult to determine how much training data is required for training raw features.

Tables 6 and 7 show performance evaluations on language embeddings based on phoneme, character and word features. These features are also compared with the baseline system. Language embeddings show an average 17% improvement on all metrics. Language embeddings based on word features achieve the best performance among the three features. Another benefit is that the linguistic feature dimension can be significantly reduced. For example, the 41,657 dimension character feature was reduced to 200 and is only 0.5% of the original dimensionality, though the EER improves about 15%.

Table 8 shows the performance of fusion systems with various combinations. We use Hungarian phonemes for phonetic-based language embeddings which attains the best performance among phoneme features. The end-to-end system used augmented data as shown in Table 5. From the result, it is clear that fusion between acoustic and language embeddings shows better efficiency than fusion between end-to-end systems such as MFCC and FBANK, although the performance on language embeddings itself shows comparably lower performance than acoustic end-to-end systems. Finally, all the scores from each feature were fused, with the fusion system using FBANK features achieving better performance than MFCC or spectrogram-based fusion systems on all measurements. Also, we find that the spectrogram system performs slightly better than the MFCC counterpart.

Although fusion of multiple systems is somewhat not feasible for a practical situation, we fused 5 different systems (i-vector with LDA and GB, FBANK based end-to-end system, language embeddings) and verified the best performance com-

Fusion system (**Bold** : end-to-end system *italic* : language embedding)	Accuracy(%)	EER(%)	C_{avg}
FBANK + *word*	76.94	13.66	13.57
FBANK + *char*	76.61	13.89	13.87
FBANK + *phoneme*	75.13	14.95	14.79
FBANK + MFCC	74.40	15.63	15.50
MFCC + *word + char + phoneme*	77.48	14.02	14.00
FBANK + *word + char + phoneme*	**78.15**	**12.77**	**12.51**
Spectrogram + *word + char + phoneme*	77.88	13.34	13.24
i-vector + **FBANK** + *word + char + phoneme*	**81.36**	**11.03**	**10.90**

Table 8: Performance measurement of score fusion systems with end-to-end system and language embeddings.

Systems	Accuracy(%)	
	Single System	Fusion System
Khurana et al. [5]	67	73
Shon et al. [9]	69.97	75.00
Najafian et al. [7]	59.72	73.27
Bulut et al. [10]	-	79.76
Our approach	**73.39**	**81.36**

Table 9: Performance comparison to previous works on same MGB-3 dataset

pared to previous studies. Table 9 shows summary performance of previous studies on same MGB-3 Arabic dialect dataset. Both single and fusion system show outstanding performance than others.

7. Conclusion

In this paper, we describe an end-to-end dialect identification system using acoustic features and language embeddings based on text-based linguistic features. We investigated several acoustic and linguistic features along with various dataset augmentation techniques using a limited dataset resource. We verified that the end-to-end system based on acoustic feature outperforms i-vectors and also that language embeddings derived from a Siamese neural network boost the performance by learning the similarities between utterances with the same dialect and dissimilarities between different dialects. Experiments on the MGB-3 Dialectical Arabic corpus show that the best single system achieves 73% accuracy, and the best fusion system shows 78% accuracy. From the experiments, spectrograms could be utilized as acoustic feature when the training dataset is large enough. We also observe that the end-to-end dialect identification system can be significantly improved using random segmentation and volume/speed perturbation to increase the diversity and amount of training data. The end-to-end dialect identification system has a simplified topology and training methodology compared to a bottleneck feature based i-vector extraction scheme. Finally, using a Siamese network to learn language embeddings reduces the linguistic-feature dimensionality significantly, and provide synergistic fusion with acoustic features.

8. References

[1] Patrick Cardinal, Najim Dehak, Yu Zhang, and James Glass, "Speaker adaptation using the i-vector technique for bottleneck features," in *Interspeech*, 2015, pp. 2867–2871.

[2] Fred Richardson, Douglas Reynolds, and Najim Dehak, "A Unified Deep Neural Network for Speaker and Language Recognition," in *Interspeech*, 2015, pp. 1146–1150.

[3] Najim Dehak, Pedro a. Torres-Carrasquillo, Douglas Reynolds, and Reda Dehak, "Language recognition via Ivectors and dimensionality reduction," in *Interspeech*, 2011, number August, pp. 857–860.

[4] Ahmed Ali, Stephan Vogel, and Steve Renals, "Speech Recognition Challenge in the Wild: ARABIC MGB-3," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2017, pp. 316–322.

[5] Sameer Khurana, Maryam Najafian, Ahmed Ali, Tuka Al Hanai, Yonatan Belinkov, and James Glass, "QMDIS : QCRI-MIT Advanced Dialect Identification System," in *Interspeech*, 2017, pp. 2591–2595.

[6] Ahmed Ali, Najim Dehak, Patrick Cardinal, Sameer Khurana, Sree Harsha Yella, James Glass, Peter Bell, and Steve Renals, "Automatic dialect detection in Arabic broadcast speech," in *Interspeech*, 2016, vol. 08-12-Sept, pp. 2934–2938.

[7] Maryam Najafian, Sameer Khurana, Suwon Shon, Ahmed Ali, and James Glass, "Exploiting Convolutional Neural Networks for Phonotactic Based Dialect Identification ," in *ICASSP*, 2018.

[8] Maryam Najafian, Saeid Safavi, Phil Weber, and Martin Russell, "Identification of British English regional accents using fusion of i-vector and multi-accent phonotactic systems," in *Proceedings of Odyssey - The Speaker and Language Recognition Workshop*, 2016, pp. 132–139.

[9] Suwon Shon, Ahmed Ali, and James Glass, "MIT-QCRI Arabic Dialect Identification System for the 2017 Multi-Genre Broadcast Challenge," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2017, pp. 374–380.

[10] Ahmet E. Bulut, Qian Zhang, Chunlei Zhang, Fahimeh Bahmaninezhad, and John H. L. Hansen, "UTD-CRSS Submission for MGB-3 Arabic Dialect Identification: Front-end and Back-end Advancements on Broadcast Speech," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2017.

[11] Ma Jin, Yan Song, Ian McLoughlin, Wu Guo, and Li Rong Dai, "End-to-end language identification using high-order utterance representation with bilinear pooling," in *Interspeech*, 2017, pp. 2571–2575.

[12] Trung Ngo Trong, Ville Hautamaki, and Kong Aik Lee, "Deep Language : a comprehensive deep learning approach to end-to-end language recognition," in *Proceedings of Odyssey - The Speaker and Language Recognition Workshop*, 2016, pp. 109–116.

[13] Arsha Nagraniy, Joon Son Chung, and Andrew Zisserman, "VoxCeleb: A large-scale speaker identification dataset," in *Interspeech*, 2017, pp. 2616–2620.

[14] David Snyder, Pegah Ghahremani, Daniel Povey, Daniel Garcia-Romero, and Yishay Carmiel, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," in *Interspeech*, 2017, pp. 165–170.

[15] Jane Bromley, James W. Bentz, Léon Bottou, Isabelle Guyon, Yann Lecun, Cliff Moore, Eduard Säckinger, and Roopak Shah, "Signature Verification Using a "Siamese" Time Delay Neural Network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 07, no. 04, pp. 669–688, 1993.

[16] Pavel Matejka, Le Zhang, Tim Ng, Sri Harish Mallidi, Ondrej Glembek, Jeff Ma, and Bing Zhang, "Neural Network Bottleneck Features for Language Identification," 2014.

[17] Suwon Shon, Seongkyu Mun, and Hanseok Ko, "Recursive Whitening Transformation for Speaker Recognition on Language Mismatched Condition," in *Interspeech*, 2017, pp. 2869–2873.

[18] Suwon Shon and Hanseok Ko, "KU-ISPL Speaker Recognition Systems under Language mismatch condition for NIST 2016 Speaker Recognition Evaluation," *ArXiv e-prints arXiv:1702.00956*, 2017.

[19] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Interspeech*, 2015, pp. 2440–2444.

[20] Hagai Aronowitz, "Inter dataset variability compensation for speaker recognition," in *IEEE ICASSP*, 2014, pp. 4002–4006.

[21] Daniel Garcia-Romero, Xiaohui Zhang, Alan McCree, and Daniel Povey, "Improving Speaker Recognition Performance in the Domain Adaptation Challenge Using Deep Neural Networks," in *IEEE Spoken Language Technology Workshop (SLT)*, 2014, pp. 378–383.

[22] Stephen Shum, Douglas a. Reynolds, Daniel Garcia-Romero, and Alan McCree, "Unsupervised Clustering Approaches for Domain Adaptation in Speaker Recognition Systems," in *Proceedings of Odyssey - The Speaker and Language Recognition Workshop*, 2014, pp. 265–272.

[23] Suwon Shon, Seongkyu Mun, Wooil Kim, and Hanseok Ko, "Autoencoder based Domain Adaptation for Speaker Recognition under Insufficient Channel Information," in *Interspeech*, 2017, pp. 1014–1018.

[24] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, pp. 1–14, 2014.

[25] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "Audio Augmentation for Speech Recognition," in *Interspeech*, 2015, pp. 3586–3589.

[26] Sameer Khurana and Ahmed Ali, "QCRI advanced transcription system (QATS) for the Arabic Multi-Dialect Broadcast media recognition: MGB-2 challenge," in *IEEE Workshop on Spoken Language Technology(SLT)*, 2016, pp. 292–298.

[27] Ahmed Ali, Yifan Zhang, Patrick Cardinal, Najim Dahak, Stephan Vogel, and James Glass, "A complete KALDI recipe for building Arabic speech recognition systems," in *IEEE Spoken Language Technology Workshop (SLT)*. dec 2014, pp. 525–529, IEEE.

[28] P. Schwarz, P. Matejka, and J. Cernocky, "Hierarchical Structures of Neural Networks for Phoneme Recognition," in *IEEE ICASSP*. 2006, vol. 1, pp. I–325–I–328, IEEE.

[29] Ahmed Ali, Peter Bell, James Glass, Yacine Messaoui, Hamdy Mubarak, Steve Renals, and Yifan Zhang, "The MGB-2 challenge: Arabic multi-dialect broadcast media recognition," in *IEEE Spoken Language Technology Workshop (SLT)*. dec 2016, pp. 279–284, IEEE.

[30] "The NIST 2015 Language Recognition Evaluation Plan", Available : https://www.nist.gov/document/lre15 evalplanv23pdf," .