NATIVE LANGUAGE IDENTIFICATION FROM RAW WAVEFORMS USING DEEP CONVOLUTIONAL NEURAL NETWORKS WITH ATTENTIVE POOLING

Rutuja Ubale, Vikram Ramanarayanan, Yao Qian, Keelan Evanini,
Chee Wee Leong, and Chong Min Lee

Educational Testing Service Research, USA
{rubale, vramanarayanan, yqian, kevanini, cleong, clee001}@ets.org

ABSTRACT

Automatic detection of an individual's native language (L1) based on speech data from their second language (L2) can be useful for informing a variety of speech applications such as automatic speech recognition (ASR), speaker recognition, voice biometrics, and computer assisted language learning (CALL). Previously proposed systems for native language identification from L2 acoustic signals rely on traditional feature extraction pipelines to extract relevant features such as mel-filterbanks, cepstral coefficients, i-vectors, etc. In this paper, we present a fully convolutional neural network approach that is trained end-to-end to predict the native language of the speaker directly from the raw waveforms, thereby removing the feature extraction step altogether. Experimental results using this approach on a database of 11 different L1s suggest that the learnable convolutional layers of our proposed attention-based end-to-end model extract meaningful features from raw waveforms. Further, the attentive pooling mechanism in our proposed network enables our model to focus on the most discriminative features leading to improvements over the conventional baseline.

Index Terms— end-to-end learning, native language identification, raw waveform processing, deep convolutional networks, attentive pooling

1. INTRODUCTION

The task of native language identification (NLI) aims at recognizing the native language (L1) of a speaker given their speech in a second language (L2). L1 information can support improved performance for speech processing tasks such as automatic speech recognition (ASR) [1, 2, 3] and speaker recognition. In addition, knowledge of a language learner's L1 can be potentially used to aid adaptive computer-assisted language learning systems in providing L1-specific training and customized feedback to the learner. Non-native speech frequently exhibits characteristic patterns due to the influence of a speaker's L1 that distinguish it from native speech, such as a distinct foreign accent, typical pronunciation errors, and patterns of intonation and duration. Capturing common characteristics that each group of L1 speakers maintains while speaking an L2 can enable adaptation and improved performance of interactive voice applications thereby improving the interaction between users and machines.

Past attempts at L1 detection from speech have relied on the use of classic front-end features such as filter banks, Mel-frequency cepstral coefficients (MFCC), and Perceptual linear predictive coefficients (PLP). In addition, features such as phoneme confusions, short term prosody-dynamics on the phone-level, and lexical features such language usage errors are popularly used for L1 detection from L2 speech. The best performing systems developed for the Computational Paralinguistics Challenge (ComParE) [4] at Interspeech 2016, which provided a venue for the comparison of multiple competing NLI systems, employed i-vector based approaches [5, 6, 7], and achieved approximately 70% or higher accuracy for identification of the 11 L1 languages. Spectrogram-based approaches for NLI built for the ComParE challenge were much inferior in performance, and only achieved accuracy rates in the range of 45%-58% [8, 9, 10]. Conventional systems for speech-based NLI consist of probabilistic models such as a Gaussian mixture model universal background model (GMM-UBM), or an i-vector framework as the front-end followed by a back-end scoring model using the cosine-similarity metric, linear discriminant analysis (LDA), or probabilistic linear discriminant analysis (PLDA). In recent years, a great deal of attention has been paid to end-to-end learning from spectral features for ASR [2, 11], spoken language understanding (SLU) [12, 13, 14] and speaker verification [15, 16]. Most recent efforts aimed at end-to-end modeling from spectral features with attention-based neural networks [17] tried to combine feature representation learning from filterbanks and predictive modeling in a single system as opposed to the traditional i-vector system where both are implemented as separate components. The proposed system achieved promising results when compared to i-vector based systems.

Deep learning has been progressively gaining recognition in numerous speech-related tasks as a viable alternative to classical front-end spectral and cepstral features such as MFCC, PLP and filter banks [18, 19, 20, 21]. Rather than assuming that these front-ends are universally applicable to multiple speech classification tasks, recent advances in neural network architectures have opened up possibilities for joint optimization of the front-end feature extraction/representation and the back-end modeling and classification targeted at ASR [22, 23, 19], speaker recognition [18] and emotion recognition [24, 25]. In addition, deep architectures that operate on raw speech signals may facilitate the learning of low-level representations that are more customized for specific tasks.

Previous work on NLI has mostly focused on i-vector based approaches, with limited efforts on spectral-based systems. However, to our knowledge, there has been no work examining NLI directly from raw waveforms. In this paper, we propose and study attention-based very deep 1-D CNN models that process time-series waveforms to predict the L1 class of the speaker from English speech. Our study shows that the best end-to-end model with fewer model parameters can outperform the conventional i-vector system by capturing speech commonalities across L1s using an attentive pooling mechanism.

2. END-TO-END NLI FROM RAW WAVEFORM

End-to-end NLI is analogous to an audio classification task that takes the time-domain L2 signal and maps the feature vector into one of the L1 classes $c \in (c_1, c_2, ..., c_n)$. We train our network to estimate the posterior probability $P(c|x)$ where $x = (x_1, x_2, ..., x_t)$ is the input signal represented as a 1-D vector. The class with the highest prediction probability is selected as the recognized L1 of the speaker.

2.1. Fully Convolutional Neural Networks

The first layer in our architecture operates over the raw time-domain waveform to perform a set of time-domain convolutions between the input waveform and some filters which can be thought of as a finite impulse-response (FIR) filterbank. The first layer is the most crucial part of waveform-based models as it deals with high-dimensional inputs. This layer is also more susceptible to vanishing gradient problems, especially when employing very deep architectures. Each convolution operation on a small window of the raw speech signal $x[n]$ is defined as:

$$y[n] = x[n] * h[n] = \sum_{l=0}^{L-1} x[l] \cdot h[n-l] \tag{1}$$

where $h[n]$ represents the filter of length L, and $y[n]$ is the filtered output. We chose a large receptive field in our first convolutional layer based on our sampling rate to mimic a bandpass filter. As our waveforms are sampled at an audio sampling rate of 16kHz, we selected a filter size of 160 to cover a duration of 10 ms. This layer is followed by convolutional layers with smaller receptive fields but a relatively larger number of filters to extract higher-level features.

Table 1 outlines the architectures of the proposed raw-waveform CNN with 5 layers (CNN-5), the very deep CNN with 18 layers (VDCNN-18), and the residual network with 34 layers (ResNet 34). We have added Batch Normalization [26] layers to prevent the gradients from getting too small or too large by normalizing data across each batch, thereby mitigating the impact of vanishing gradient problems commonly encountered during training of very deep architectures. We do not add any dropout layers as batch normalization can provide a regularizing effect to reduce overfitting and speed up the convergence of the network. After batch normalization, the output of each filter is passed through a ReLU activation (a half-wave rectifier) to lower the computational cost. Max-pooling layers with stride of 4 are applied to reduce the dimensionality of the feature maps and extract invariant features.

We adopt the following design strategies in the convolutional layers: (i) for the same output feature map size, the layers have the same number of filters; and (ii) when the feature map size is reduced by a factor of 4 via max pooling, the number of filters is doubled in order to preserve the computational complexity per layer. Our design of a fully convolutional network without any fully connected layers will have far fewer parameters when compared to a similar network that includes fully connected layers. This is also substantiated by [27] which indicated that fully connected layers may not be required in a neural network that processes raw audio data as the input. Additionally, [28] make a strong case for the use of convolutional layers instead of fully connected layers in a neural network that processes raw audio data as the input. At the end of our network we use a Global Average Pooling (GAP) layer that averages the activations along the time dimension followed by a softmax layer that generates the probabilities for each L1 class.

Table 1. Architectures of proposed fully convolutional networks. In the table every layer is represented as {filter width, # filters} while × represents number of layers of same configuration

CNN-5	**VDCNN-18**	**ResNet 34**
$\{160, 128\}$	$\{160, 64\}$	$\{160, 48\}$
Maxpooling: 4×1		
$\{3, 128\}$	$\{3, 128\} \times 4$	$\left\{\begin{array}{c}3, 48\\3, 48\end{array}\right\} \times 3$
Maxpooling: 4×1		
$\{3, 256\}$	$\{3, 256\} \times 4$	$\left\{\begin{array}{c}3, 96\\3, 96\end{array}\right\} \times 4$
Maxpooling: 4×1		
$\{3, 512\}$	$\{3, 256\} \times 4$	$\left\{\begin{array}{c}3, 192\\3, 192\end{array}\right\} \times 6$
Maxpooling: 4×1		
	$\{3, 512\} \times 4$	$\left\{\begin{array}{c}3, 384\\3, 384\end{array}\right\} \times 3$
Global Average/ Attentive Pooling		
Softmax		

2.2. Residual Learning

We experiment with ResNets [29] to investigate the performance of training deeper networks with up to 34 layers. Generally, in a traditional network the layers are trained to learn the true distribution $H(x)$, where x is the input to the layers. In a residual block, layers are approximated to learn the residual mapping, $R(x)$, where $R(x) = H(x) - x$. In residual networks (ResNets), residuals are expressed via skip connections between layers as shown in Figure 1. Residual mapping speeds up learning by reducing the impact of vanishing gradients, as there are fewer layers to propagate through.

2.3. Attentive pooling

Attention in CNNs has been mainly implemented as attentive pooling, i.e., applied to the pooling layer. Simple global pooling methods like average or max pooling are typically adopted in fully convolutional networks. Using these pooling methods, feature maps can be efficiently projected to 1-dimensional vectors. However, as these methods treat all input data equally, i.e., they weigh the contribution of each instance equally, it is possible that they tend to skip some salient features, thereby potentially affecting the quality of the prediction results. Within time-domain waveforms, certain regions may be more valuable in predicting L1s. Thus, we experiment with an attention-based pooling mechanism in our architecture to achieve more optimal prediction. We adopt the strategy similar to that proposed in other speech processing tasks [30, 31, 32]. Let $x = (x_1, x_2, ..., x_T)$ represent the high-level feature representation at the output of the final CNN layer before global average pooling where T is the length of x. We compute the importance weight e_i for the feature representation x_i using the equation:

$$e_i = v^T \tanh(Wx_i + b), \qquad i = 1, ..., T \tag{2}$$

We then use a softmax function to compute the normalized

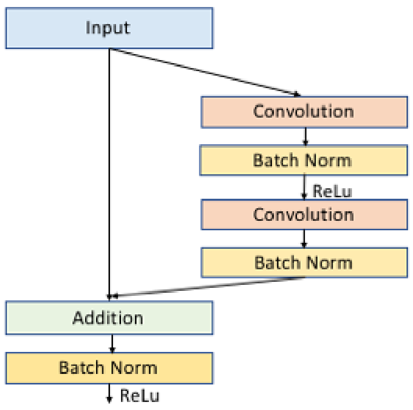

Fig. 1. Residual block used in ResNet-34

weight α_i as:

$$\alpha_i = \frac{\exp(e_i)}{\sum_{k=1}^{T}\exp(e_k)} \tag{3}$$

The normalized score is then used as the weight in the pooling layer to compute the weighted average vector:

$$\tilde{\mu} = \sum_{i=1}^{T} \alpha_i x_i \tag{4}$$

The weighted average vector focuses on the elements that are of most importance thereby making the vector representation more L1-discriminative.

3. EXPERIMENTS

3.1. Corpora

We use a corpus of non-native English speech collected during a high-stakes global assessment of English language proficiency. This corpus is a more comprehensive version of the Native Language sub-challenge corpus used for ComParE Challenge at Interspeech 2016 [4]. The corpus consists of spoken responses from 11,000 non-native speakers with 11 different L1 backgrounds: Arabic (ARA), Chinese (CHI), French (FRE), German (GER), Hindi (HIN), Italian (ITA), Japanese (JAP), Korean (KOR), Spanish (SPA), Telugu (TEL) and Turkish (TUR). Each response is approximately 45-60 seconds long. The [17] contains approximately 138 hours of speech sampled at 16 kHz. There are approximately 1,000 speech recordings for each L1 in the dataset. The data is partitioned as follows with no overlapping speakers: 7,040 recordings are used for training, 1,760 recordings for validation, and 2,200 recordings for evaluating performance of our models. The test set contains 200 responses for each L1.

3.2. Experimental Setup

3.2.1. End-to-end raw waveform models

We used very long speech sequences of 45 seconds in duration (approximately 720,000 samples at 16kHz) as input to the deep networks as a majority of the recordings in the corpus are around 45 seconds long. The waveforms are mean-variance normalized at the utterance level. We tried different speech sequence lengths ranging from 10s to 45s and achieved the best performance by using the full 45-second segments. The architectures for the raw waveform models are shown in Table 1. The kernel weights in each CNN layer of all fully convolutional networks are initialized using a Glorot uniform distribution [33] and regularized using l_2-regularization. All end-to-end raw waveform models are trained using Keras with a Tensorflow backend on two CUDA-enabled GPUs. The networks are trained for 200 epochs with a batch size of 32 samples. All waveform models are trained using categorical cross-entropy as the cost function and optimized using a Stochastic Gradient Descent (SGD) optimizer with drop-based learning rate decay and momentum of 0.8. An initial learning rate of 0.1 is used in our experiments and the learning rate decay function is implemented such that the learning rate (lr) is dropped by one half every 10 epochs:

$$lr = intial\ lr * drop^{floor(epoch/epoch\_drop)} \tag{5}$$

We present our experiments with 5, 18 and 34 layers in Section 4. Overall, we experimented with different numbers of layers, numbers of filters, and filter sizes. The architectures with the parameters reported in the paper achieved the highest performance rates compared to the other models that were investigated.

3.2.2. End-to-end filterbank models

We compare the performance of Mel-filterbank features in neural networks and raw waveforms. [17] proposed an attention-based three layer pyramidal Bidirectional Gated Recurrent Unit (pBGRU) architecture referred to as Listen, Attend and Identify (LAI), an architecture inspired from the state-of-the-art ASR model in [22]. We use this model as a baseline for comparison. The Listen encoder consists of three pBGRU layers with 512 nodes i.e. 256 nodes in each direction. The weights in all pBGRU layers are initialized using the Glorot-Bengio initializer and Hyperbolic Tangent (tanh) activation. A dropout layer with a dropout rate of 30% is added after each pBGRU layer to avoid overfitting. The acoustic encoder is followed by the identifier, an attention-based classifier that applies attention to the output of the listener. The attention layer is followed by a fully connected feed-forward layer with softmax activation with a hidden size of 11 corresponding to the 11 L1 classes. The model is trained using categorical cross-entropy criterion and optimized using Adam optimizer [34] with a learning rate of 0.0005.

For consistency with the CNN experiments, we also include results in Section 4 from a CGDNN model [17, 11] that consists of a combination of 2D CNN layers and Gated Recurrent Units (GRU) to perform temporal and frequency modeling followed by two fully connected layers. The first layer is a 2D convolution layer with 16 output filters in the convolution, kernel size of 7×7 followed by a max pooling layer of pooling size 6×6. The second 2D CNN layer has kernel size of 5×5 and 32 output filters. The next two CNN layers have kernel size of 3×3 and 32 output filters. The last three CNN layers are each followed by a max pooling layer of pooling size 3×1. All CNN layers use Rectified linear unit (ReLU) activation. The CNN network is connected to a linear layer with 128 nodes to reduce the dimensionality of the feature vector. This is followed by two GRU layers each with 256 nodes. The last GRU layer is followed by an attention layer connected to a feed-forward layer with 32 nodes followed by a fully connected softmax layer with 11 output nodes.

Results of a simple CNN network with the first 4 CNN layers of the CGDNN model which formed a baseline in [17] are also shown in Section 4. Adding more layers did not improve the performance of the CNN architecture. The CGDNN and CNN networks are both trained using an SGD optimizer with a drop-based learning rate decay as shown in Equation 5. All the models described in this section use a cross-layer attention mechanism that yielded the best performance in [17] where the attention vector is computed as the weighted average of the final layer outputs of the encoder network but with weights computed using the second-to-last layer. [17] provides complete architecture details for these systems.

3.2.3. *i-vector system*

A GMM-based i-vector system is used as the baseline NLI system. Here an energy-based voice activity detection (VAD) method is applied to detect non-speech segments within utterances. 20-dimensional MFCCs including c_0 were extracted from the resultant speech segments via a 20ms Hamming window with a 10ms time shift. MFCCs are appended with their first and second derivatives. Utterance-based cepstral mean normalization was performed on the acoustic feature vectors. A GMM with 2,048 Gaussian kernels and a full covariance matrix was trained as the Universal Background Model (UBM) by using the corpus described in Section 3.1 for training. The same training set was also used to train an i-vector extractor T-matrix, i.e., a low rank rectangular matrix called the total variability matrix, as well as Probabilistic Linear Discriminant Analysis (PLDA) projection matrices. We employed PLDA as a scoring method for L1 recognition, where we calculated the log likelihood rate (LLR) for the i-vector of each testing utterance and those of target L1s and select the one with the highest LLR as the recognized L1.

3.2.4. *x-vector system*

X-vectors, or DNN-based speaker embeddings, are now trending as state-of-the-art systems in speaker recognition. We present preliminary results using the Kaldi x-vector system recipe [35]. The input features to the DNN are 30-dimensional filterbanks using a 25ms Hamming window. 30 MFCCs were calculated every 10ms and this feature vector was mean-normalized over a sliding window of up to 3 seconds. We used the energy-based VAD as used in the i-vector baseline in Section 3.2.3 to filter out non-speech frames. The DNN architecture is depicted in Table 2. The first part of the DNN contain 5 layers of a time-delay architecture that operates at the frame-level. The first 4 layers contain 512 neurons each and the fifth layer contains 1500 neurons. This is followed by a statistics pooling layer which the computes its mean and standard deviation over all frame-level outputs. A single vector of aggregated means and standard deviations is propagated through two hidden layers with ReLU non-linearity consisting of 512 neurons each and a softmax output layer. The embeddings are extracted from the sixth layer of the DNN framework and are centered and projected using LDA with dimensionality equal to 150. After dimensionality reduction, the representations are length-normalized and modeled by PLDA to compute the LLR for each test utterance similar to the scoring method mentioned in Section 3.2.3.

3.2.5. *Fusion of end-to-end models and i-vector baselines*

[17] demonstrated that filterbank-based neural network models are able to learn complementary representations from speech data when compared to i-vector system. We investigate whether raw-waveform models are able to capture different representations and if a combination of models can yield higher performance in this work. We performed score-level fusion of posterior probabilities produced by the end-to-end models and normalized LLRs (by z-score) generated by the PLDA scoring model using a Multilayer Perceptron (MLP) classifier to predict the L1 classes. The fusion network consists of 2 fully connected layers each with 200 hidden units and ReLU activation connected to a softmax layer with 11 hidden units. The network is optimized using an SGD optimizer with an initial learning rate of 0.001, momentum of 0.9, and Nesterov's Accelerated Momentum update [36, 37]. The fusion model is trained using the validation set mentioned in Section 3.1.

Table 2. The embedding DNN configuration. x-vectors are extracted at layer segment6, before the nonlinearity. The L in the softmax layer corresponds to the number of L1s

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	150×512
frame2	$\{t-2, t, t+2\}$	9	1536×512
frame3	$\{t-3, t, t+3\}$	15	1536×512
frame4	$\{t\}$	15	512×512
frame5	$\{t\}$	15	512×1500
stats pool	$[0, T)$	T	$1500T \times 3000$
segment6	$\{0\}$	T	3000×512
segment7	$\{0\}$	T	512×512
softmax	$\{0\}$	T	$512 \times L$

Table 3. The accuracy performance (%) of different raw waveform models on the test set of the 11 L1 corpus

Method	w/o attentive pooling	attentive pooling
CNN-5	65.34	65.51
ResNet 34	75.55	78.05
VDCNN-18	78.68	80.73

4. RESULTS AND DISCUSSION

Table 3 illustrates the performance in terms of accuracy of different raw waveform-based CNN models with and without attentive pooling on the test of 11 L1s. We observed that attentive pooling contributed to significant performance improvements in ResNet 34 and VDCNN-18 models and a minor improvement in CNN-5 model. The performance of VDCNN-18 was improved from 78.68% to 80.73% with attentive pooling while the performance of ResNet 34 was improved from 75.55% to 78.05% with attentive pooling. Our proposed VDCNN-18 model with attentive pooling achieved the highest accuracy of 80.73%. The VDCNN-18 network performs better than ResNet 34 and therfore indicates that adding more layers does not help improve performance. From our experiments with different number of layers ranging from 3 to 18, we noticed that the performance improved linearly with an increase in layers and the best performance was attained at 18 layers.

We compare the performance of different baseline models discussed in Section 3.2 with our best proposed models in Table 4. The performance of the NLI systems are evaluated using accuracy and Unweighted Average Recall (UAR), which was the metric used for the 2016 ComParE challenge. The preliminary results using the x-vector framework (51.86%) show that its performance is much inferior to that of the i-vector system (79.72%). Therefore we use the

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	**158**	2	3	2	1	3	3	2	6	0	6
CHI	0	**150**	0	0	0	0	1	2	0	0	0
FRE	20	14	**182**	29	2	18	13	12	8	2	9
GER	0	0	3	**162**	0	1	0	1	0	0	4
HIN	0	0	0	0	**129**	0	0	2	1	44	0
ITA	12	5	7	3	2	**165**	4	0	5	1	5
JPN	1	9	1	0	0	0	**157**	11	4	0	3
KOR	1	16	0	0	0	2	17	**165**	6	0	4
SPA	4	2	2	3	1	11	4	3	**169**	2	3
TEL	1	1	1	0	64	0	0	0	1	**151**	0
TUR	3	1	1	1	1	0	1	2	0	0	**166**

Fig. 2. Confusion matrix of the i-vector baseline system results on the test set of 11 L1s (rows: references; column: hypotheses)

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	**147**	0	5	1	1	6	7	1	2	1	6
CHI	2	**178**	3	2	1	2	4	4	0	0	1
FRE	9	0	**154**	9	2	13	3	3	14	2	2
GER	2	1	9	**176**	3	4	2	1	4	0	5
HIN	4	0	0	1	**128**	2	0	0	1	34	1
ITA	6	2	14	1	0	**143**	2	0	4	1	7
JPN	5	3	5	1	2	1	**157**	9	5	3	3
KOR	5	12	0	4	1	7	12	**174**	12	2	2
SPA	2	1	4	2	0	15	10	3	**149**	1	3
TEL	1	0	1	0	61	1	0	0	1	**156**	1
TUR	17	3	5	3	1	6	3	5	8	0	**169**

Fig. 3. Confusion matrix of raw-waveform VDCNN-18 results on the test set of 11 L1s (rows: references; column: hypotheses)

i-vector system as our baseline system. X-vectors are generally seen to show significant improvements for speaker and language recognition task with data augmentation as seen in [38, 39]. Specifically for the task of language recognition, [38] demonstrated that embeddings computed using bottle-neck features extracted from ASR DNN as input to the x-vector framework show significant improvements over the MFCC features. To limit the scope of this paper, we will explore the performance with data augmentation and other input features to the x-vector DNN in future work.

VDCNN-18 with attentive pooling outperforms all the baseline models. VDCNN-18, both with and without attentive pooling performs substantially better than the best previously published filter-bank model, LAI, indicating that our fully convolutional raw-waveform model is able to extract more discriminative feature representations for L1 recognition. With the integration of attentive pooling, VDCNN-18 is able to show approximately 1% improvement over the i-vector baseline.

Confusion matrices for the results on the test set of 11 L1s using the i-vector baseline, VDCNN-18 without and with attentive pooling are shown in Figures 2, 3 and 4, respectively. The most separable L1s for the i-vector system are French (FRE), Spanish (SPA) and Turkish (TUR) which can achieve F1 scores 0.91, 0.845, 0.83 respectively.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	**153**	0	7	0	2	5	6	0	4	2	7
CHI	3	**178**	3	3	2	3	4	4	2	0	2
FRE	9	0	**167**	3	2	9	1	3	7	1	4
GER	0	1	4	**178**	3	5	1	1	8	1	2
HIN	6	0	3	1	**128**	0	0	0	2	34	1
ITA	3	1	6	8	2	**153**	1	0	5	2	3
JPN	3	2	1	1	1	3	**162**	7	7	2	2
KOR	2	15	1	3	0	4	19	**180**	5	1	3
SPA	5	1	4	1	0	11	4	3	**155**	2	7
TEL	4	0	0	0	59	1	1	0	1	**154**	1
TUR	12	2	4	2	1	6	1	2	4	1	**168**

Fig. 4. Confusion matrix of raw-waveform VDCNN-18 with attentive pooling results on the test set of 11 L1s (rows: references; column: hypotheses)

Table 4. The accuracy performance (%) of different NLI systems on the test set of 11 L1 corpus

Method	**Acc (%)**	**UAR (%)**	**Parameters**
Majority vote	9.00	8.26	
x-vector system	51.86	51.81	$\sim 11M$
2D CNN (fbank)	60.45	61.13	$\sim 0.63M$
CGDNN (fbank)	70.18	70.80	$\sim 0.73M$
LAI (fbank)	71.72	71.42	$\sim 4.46M$
VDCNN-18 (wave)	78.68	78.86	$\sim 3.69M$
i-vector baseline	79.72	81.59	$\sim 32M$
VDCNN-18 (attn)	80.73	80.81	$\sim 3.94M$

In particular, VDCNN-18 shows strong improvements on Chinese (CHI), German (GER), Korean (KOR) and slightly better performance for Turkish (TUR), with F1 scores of 0.89, 0.88, 0.87 and 0.845, respectively. For Turkish recognition, VDCNN-18 performs slightly better than the i-vector system. Both systems seem to perform similarly worse for Hindi (HIN) recognition but perform better on Telugu recognition (TEL). Hindi can be easily confused with Telugu as the two languages share many similarities in pronunciation. While the i-vector system is seen to outperform VDCNN-18 without attentive pooling for Arabic (ARA), French (FRE), Italian (ITA) and Spanish (SPA) recognition, we observed that attentive pooling in VDCNN-18 supports improvements for recognition of these L1s. We notice, that with attentive pooling, VDCNN-18 continues to perform consistently on recognition of L1s that were most distinguishable without attentive pooling and further improves the recognition of the under-performing L1s, and hence thereby improves the overall prediction performance.

We also report the number of parameters for the models in Table 4. With reasonably fewer parameters, our best raw waveform model is able to outperform the i-vector baseline. The VDCNN-18 model has approximately 8 times fewer parameters than the i-vector system. Our filter-bank-based CGDNN and CNN models have 5 times and 6 times fewer parameters than VDCNN-18, however they perform around 10% and 20% worse respectively than VDCNN-18 and i-vector systems indicating waveform based VDCNN-18 model is better suited to the NLI task.

Since we observe from Figures 2, 3 and 4 that VDCNN-18 per-

Table 5. The F1-score of individual L1s on 11 L1s recognition.

L1s	i-vector Baseline	VDCNN-18 (w/o attn pool)	VDCNN-18 (w/ attn pool)
ARA	0.79	0.725	0.765
CHI	0.75	0.88	0.89
FRE	0.91	0.75	0.835
GER	0.81	0.865	0.89
HIN	0.645	0.69	0.64
ITA	0.825	0.74	0.765
JPN	0.785	0.815	0.81
KOR	0.825	0.83	0.9
SPA	0.845	0.735	0.775
TEL	0.755	0.78	0.77
TUR	0.83	0.81	0.84

Table 6. The accuracy obtained by different fusion systems on the test set of 11 L1 corpus

Fusion system	**Accuracy (%)**
LAI + i-vector	82.27
CGDNN + i-vector	83.14
LAI + CGDNN + i-vector	83.32
VDCNN-18 + i-vector	85.36
VDCNN-18 + attentive pooling + i-vector	86.05

forms better for a complementary set of languages as compared to the i-vector system, we demonstrate the results of a fusion of the i-vector system with VDCNN-18 in Table 6 to see if we can get additional improvements. Fusion of VDCNN-18 with and without attentive pooling and the i-vector system yielded 85.36% and 86.05% accuracy, respectively. Also the fusion result of VDCNN-18 (attentive pooling) + i-vector system is approximately 3% better than LAI + CGDNN + i-vector.

[20] proposed an analysis of the weights learned in the first layer of a fully connected DNN that used raw waveforms as input. We follow their approach to visualize the spectral properties of the learned filters in the first convolutional layer of the raw waveform CNN. Here the weight vector of each convolutional unit corresponds to a mirrored impulse response of a filter with finite impulse response (FIR). We perform Discrete Fourier Transform (DFT) of the weights of each filter $w_{(i,\bullet)} \in \mathbb{R}^k$ to obtain the the magnitude spectrum of the filter response as follows:

$$W_i = 20 \cdot log_{10}|DFT\{w_{(i,\bullet)}\}| \qquad 0 \leq i < K \tag{6}$$

where K corresponds to the number of convolutional filters. The center frequency is estimated based on the position of the maximum in the magnitude spectrum [20, 40] as:

$$f_c^i = \underset{1\leq j\leq 8000}{\text{argmax}} \{W_{i,j}\} \tag{7}$$

The rows in W_i are reordered by the estimated filter center frequency such that $f_c^{\Pi(0)} \leq f_c^{\Pi(1)} \leq ... \leq f_c^{\Pi(K-1)}$, where $\Pi(i)$ represents the permutation of the filter indices. Figure 5 shows the corresponding magnitude spectra of the learned filters ordered by the center frequencies. The number of band pass filters is relatively high in the lower frequency region and the bandwidth of the filters becomes larger with increasing center frequency. The distribution of center frequencies is also non-linear. It can be seen from this representation that CNN has learned band-pass like filters that resemble the audiological distribution that is implemented in classical front-end feature extraction pipelines like MFCC, PLP, etc.

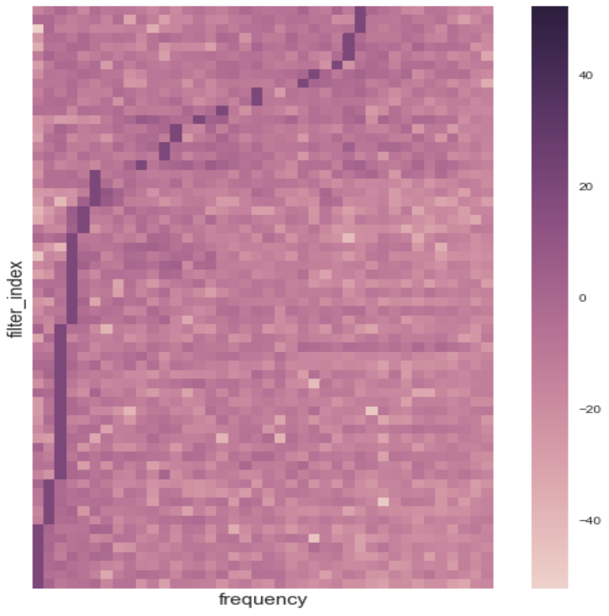

Fig. 5. Spectrum visualization of learned filters (ordered by the center frequencies) in first CNN layer of the VDCNN-18 network. Y-axis represents the index of the filter ranging from 1 to 64, and x-axis represents the frequency ranged from 0 to half the sampling rate i.e. 8kHz. Each row in the heatmap (y-axis) represents one filter and each column, a frequency interval of 205Hz (ranging from 0 to 8kHz on the x-axis).

5. CONCLUSION

In this paper, we experimented with different end-to-end fully convolutional architectures for automatic L1 recognition from time-series waveform. The proposed models are fully convolutional and are trained end-to-end to determine the L1 class of the speaker from the raw waveform. Our results demonstrate that all our raw waveform-based networks can perform significantly better than the models that use filterbank features as input. In particular, our best proposed attention-based VDCNN-18 model can outperform the traditional i-vector system by achieving 80.73% accuracy with far fewer parameters in the model. This indicates that our model is able to learn feature representations that are more discriminative and better than classic log Mel-filterbanks or i-vectors. The best proposed raw-waveform model is optimized for size with 8 times fewer parameters than the i-vector system. Further, our experimental results also show a significantly improved performance of end-to-end models with an attentive pooling mechanism in the framework. Finally, fusing the i-vector system with the end-to-end waveform system leads to a further significant improvement. Future work will involve designing more compact models with fewer model parameters for L1 recognition than those discussed in this work and exploring ways to enhance the performance of our proposed models further. We will also explore an extension of our very deep CNN networks to handle L1 recognition of more languages using raw waveforms.

6. REFERENCES

[1] C. Huang, E. Chang, J. Zhou, and K. F. Lee, "Accent modeling based on pronunciation dictionary adaptation for large vocabulary mandarin speech recognition," in *In Sixth International Conference on Spoken Language Processing*. IEEE, 2000, pp. 803–806.

[2] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, and K. Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4749–4753.

[3] K. Zechner, D. Higgins, R. Lawless, Y. Futagi, S. Ohls, and G. Ivanov, "Adapting the acoustic model of a speech recognizer for varied proficiency non-native spontaneous speech using read speech with language-specific pronunciation difficulty," in *In Tenth Annual Conference of the International Speech Communication Association*, 2009.

[4] B. W. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A.C. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception, Sincerity & Native language," in *INTERSPEECH*, 2016, vol. 2016, pp. 2001–2005.

[5] P. G. Shivakumar, S. N. Chakravarthula, and P. G. Georgiou, "Multimodal fusion of multirate acoustic, prosodic, and lexical speaker characteristics for native language identification," in *INTERSPEECH*, 2016, pp. 2408–2412.

[6] G. Gosztolya, T. Grsz, R. Busa-Fekete, and L. Toth, "Determining native language and deception using phonetic features and classifier combination," in *INTERSPEECH*, 2016, pp. 2418–2422.

[7] A. Abad, E. Ribeiro, F. Kepler, R. F. Astudillo, and I. Trancoso, "Exploiting phone log-likelihood ratio features for the detection of the native language of non-native english speakers," in *INTERSPEECH*, 2016, pp. 2413–2417.

[8] Y. Jiao, M. Tu, V. Berisha, and J. M. Liss, "Accent identification by combining deep neural networks and recurrent neural networks trained on long and short term features," in *INTERSPEECH*, 2016, pp. 2388–2392.

[9] G. Keren, J. Deng, J. Pohjalainen, and B. W. Schuller, "Convolutional neural networks with data augmentation for classifying speakers' native language," in *INTERSPEECH*, 2016, pp. 2393–2397.

[10] A. Rajpal, T. B. Patel, H. B. Sailor, Patil H. A. Madhavi, M. C., and H. Fujisaki, "Native language identification using spectral and source-based features," in *INTERSPEECH*, 2016, pp. 2383–2387.

[11] T.N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4580–4584.

[12] Y. Qian, R. Ubale, V. Ramanaryanan, P. Lange, D. Suendermann-Oeft, K. Evanini, and E. Tsuprun, "Exploring ASR-free end-to-end modeling to improve spoken language understanding in a cloud-based dialog system," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 569–5764.

[13] Y. Qian, R. Ubale, P. Lange, K. Evanini, and F. Soong, "From speech signals to semantics - tagging performance at acoustic, phonetic and word levels," in *Chinese Spoken Language Processing (ISCSLP), 2018 11th International Symposium (to appear)*. IEEE, 2018.

[14] D. Serdyuk, Y. Wang, C. Fuegen, A. Kumar, B. Liu, and Y Bengio, "Towards end-to-end spoken language understanding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018.

[15] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[16] S. X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification.," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 171–178.

[17] R. Ubale, Y. Qian, and K. Evanini, "Exploring end-to-end attention-based neural networks for native language identification," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 84–91.

[18] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 5509–5513.

[19] N. Zeghidour, N. Usunier, I. Kokkinos, T. Schaiz, G. Synnaeve, and E. Dupoux, "Learning filterbanks from raw speech for phone recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5509–5513.

[20] Z. Tuske, P. Golik, R. Schluter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for lvcsr," in *Proc. Interspeech*, 2014.

[21] T.N. Sainath, B. Kingsbury, A. Mohamed, and B. Ramabhadran, "Learning filter banks within a deep neural network framework," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013.

[22] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[23] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4845–4849.

[24] M. Sarma, P. Ghahremani, D. Povey, N. K. Goel, K. K. Sarma, and N. Dehak, "Emotion identification from raw speech signals using dnns," in *Interspeech*, 2018, pp. 3097–3101.

[25] S. Albanie, A. Nagrani, A. Vedaldi, and A. Zisserman, "Emotion recognition in speech using cross-modal transfer in the wild," in *Proceedings of the 26th ACM International Conference on Multimedia*, 2018, pp. 292–301.

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*.

[27] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, "Very deep convolutional neural networks for raw waveforms," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 421–425.

[28] F. Longueira and S. Keene, "A fully convolutional neural network approach to end-to-end speech enhancement," in *arXiv preprint arXiv:1807.07959*, 2018.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2016, pp. 770–778.

[30] Y. Zhang, J. Du, Z. Wang, J. Zhang, and Y. Tu, "Attention based fully convolutional network for speech emotion recognition," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2018, pp. 1771–1775.

[31] T. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," arXiv preprint arXiv:1807.07959, 2018.

[32] Xugang Lu, Peng Shen, Sheng Li, Yu Tsao, and Hisashi Kawai, "Temporal attentive pooling for acoustic event detection," in *Proc. Interspeech*, 2018, pp. 1354–1351.

[33] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015, vol. 5.

[35] https://github.com/kaldi asr/kaldi/blob/master/egs/voxceleb/v2, ," .

[36] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$," in *Doklady AN SSSR (translated as Soviet. Math. Docl.)*, 1983, vol. 269, pp. 543–547.

[37] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, "Advances in optimizing recurrent networks," arXiv preprint arXiv:1212.0901, 2012.

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

[39] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[40] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *In Sixteenth Annual Conference of the International Speech Communication Association*, 2015.