End-to-end DNN-CNN Classification for Language Identification

Ma Jin, Yan Song, and Ian McLoughlin

Abstract—**A defining problem in spoken language identification (LID) is how to design effective representations which allow features to be extracted that are specific to language information. Recent advances in deep neural networks for feature extraction have led to significant improvements in results, with deep end-to-end methods proving effective. In this paper, a novel network is proposed and explored that models an effective representation using first and second-order statistics of features extracted from a well-trained phoneme-related DNN bottleneck network followed by a stack of CNN convolutional layers. The high-order statistics extracted through second order pooling at the output of the CNN are robust to speaker and channel variability, and background noise. Evaluation with NIST LRE 2009 shows improved performance compared to current state-of-the-art systems, achieving over 33% and 20% relative equal error rate (EER) improvement for 3s and 10s utterances.**

Index Terms—**anguage identification, utterance representation extraction, convolutional neural network, deep neural network, bilinear pooling**

I. Introduction

ONE of the key issues in building language identification (LID) classifiers is how to extract efficient and compact features that are specific to LID information, and useful for discriminating between languages. This is challenging due to large variation in speech content, speakers, channels and background noise, coupled with a scarcity or mismatch in training resources. Total variability (TV) methods currently tend to achieve state-of-the-art performance through their powerful ability to model, exploiting zeroth, first and second order Baum-Welch statistics of features in a speaker, phoneme and channel dependent space. This is true both in speaker recognition (SR) [1] and language identification (LID) [2] domains. However, i-vectors are extracted in an unsupervised fashion and consequently need discriminant backends such as Linear Discriminant Analysis (LDA) and Within-Class Covariance Normalization (WCCN). Due to the generative attributes of Gaussian Mixture Models (GMM), it is more difficult to model the variance of short speech utterances, thereby significantly reducing performance compared to long utterances.

Deep learning techniques have been shown to achieve impressive results in related applications that include large scale speech recognition and image classification. Deep Neural Networks (DNN) demonstrate particularly strong learning capabilities in both front-end feature extraction and back-end modelling positions. For example, Song *et.al*, Richardson *et.al* and Jiang *et.al* [3], [4], [5] proposed the use of deep bottleneck features (DBFs) from a well trained DNN for automatic speech recognition (ASR) [6]. DBFs – formed from a lower dimensionality central construction in a trained deep network, are inherently quite robust to phonotactically irrelevant information. DNNs have been shown to excel when combined with phonotactic training in LID modelling, nevertheless both the DBFs and their calculated statistics are extracted from phoneme information. Neither phonemes or phoneme states are discriminative between all combinations of languages.

To extract language discriminant features and representations, more and more end-to-end NNs have been proposed which can span from a frame level to an utterance level LID identity – avoiding the need for separate back-end algorithms which are discriminative. End-to-end schemes have been used in image processing [7], [8], [9] and speech recognition [10], combining good performance with convenience in training. Lopez-Moreno *et.al* [11] proposed an end-to-end LID scheme that used large scale DNNs, and which which had good performance. In their scheme, speech is segmented into small parts containing just a few frames, with each part aligned into a specific language ID. However it can be difficult to train a language discriminant model because DNN input dimension may not scale to the size necessary to represent a language discriminant unit. Garcia-Romero *et.al* [12] improved upon this by introducing the use of a time delay neural network (TDNN), which spans a wider temporal context, hence contributing a greater number of features from which to form statistics. In that approach, a bottom-up hierarchical structure was used to produce a posterior probability over the set of languages concatenated over a long time span. Gelly *et.al* [13] and Gonzalez *et.al* [14] proposed building Long Short Term Memory-Recurrent Neural Networks (LSTM-RNN) to identify languages. This architecture has natural advantages of sequence modelling which can choose what to remember and to forget automatically across a wide context. Geng *et.al* [15] applied attention-based RNN mechanisms, first used in neural machine translation, to LID. In that scheme, each speech frame had a posterior, forming vectors that were weighted and summed into a single utterance representation. The unified nature of that architecture allowed it to benefit from end-to-end training, which boosted system performance.

Unlike LSTM-RNNs, convolutional neural networks (CNN) tend to be more flexible and hence many variant architectures have been published [18], [17], [16]. Aiming for similar performance gains, the authors [19] also introduced an end-to-end approach, which was named LID-net. This combined the proven frame-level feature extraction capabilities of DNNs with the effective utterance level organising abilities of CNNs. In that network, language discriminant features were obtained in intermediate CNN layers. We

Manuscript received April 9, 2017; revised April 17, 2017.

Ma Jin and Yan Song are with the National Engineering Laboratory of Speech and Language Information Processing, Anhui, China. e-mail: jinma525@mail.ustc.edu.cn.

Ian McLoughlin is with the University of Kent School of Computing, Medway, UK.

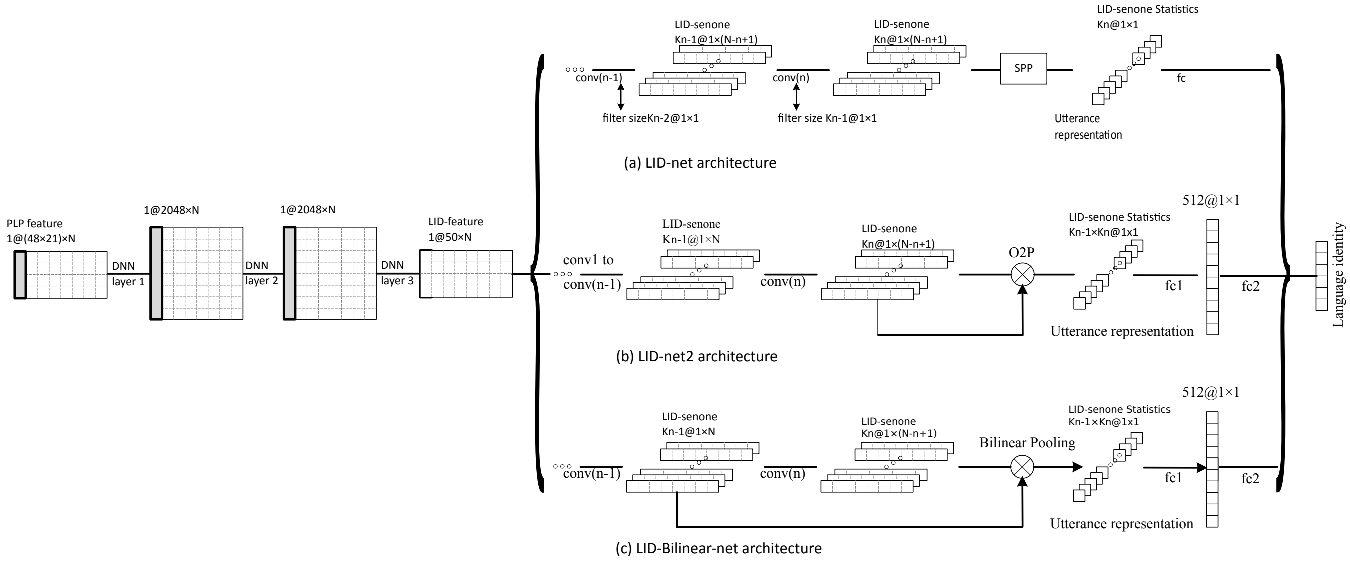

Fig. 1. (a)**LID-net** (top) where features are extracted frame-by-frame from *DNN layers 1-3*. LID-senones are obtained through several convolutional layers, with the expansion of filter size in convolutional layer 1 to a context of 21 frames, followed by several 1×1 filters (convolutional layers 2 to n). (b) **LID-net2** (middle) is based on LID-net up to the pooling layer which is the second order pooling of the LID-senones, yielding higher order statistics. (c) **LID-bilinear-net** (bottom) based on LID-net2 but now the statistics are derived from the outer product of two feature maps from different convolutional layers, from which first and second order statistics can be obtained.

named the features LID-senones because they were analogous to the senones used in many competing methods. The initial performance evaluation of LID-net showed that it was at least as good as state-of-the-art DBF/i-vector systems, and slightly better for short utterances. While it worked well, LID-net only averaged LID-senone posteriors using zeroth order Baum-Welch statistics and did not make use of higher level statistics.

In this paper, we extend the idea further by extracting first and second order Baum-Welch statistics. The method we propose is inspired by the image processing domain where two dimensional feature maps are common. Perronnin *et.al* and Carreira *et.al* introduced fisher vector (FV) [20] and second order pooling (O2P) [21] respectively, showing how first and second order statistics, widely used in patten recognition, can be extracted and contribute outstanding performance to classification. In LID-net2, we use O2P to deriver utterance-level feature statistics on LID-senones, and will show that performance is excellent. However DNN and CNN intermediate layers are known to form graduated representations from the input features to the output classes. In this case, the CNN transforms LID-features into language classes. Since the layers concentrate information differently, there is potential for output from multiple layers to be somewhat complimentary. We therefore test a system that generates the higher order statistics from different layers. This is called LID-bilinear-net. In both LID-net2 and LID-bilinear-net, we include additional fully-connected (fc) layers for the output classification to map the pooled statistics into the language classes.

A. Contribution

We introduce two end-to-end DNN-CNN neural network variants that utilize high-order LID-senone statistics. Both systems combine the advantage of both the high-order Baum-Welch statistics calculation of i-vector systems with the natural discriminant attributes of neural networks. In **LID-net2**, high-order statistics are obtained through an O2P method borrowed from fine-grained visual recognition [22], whereas in **LID-bilinear-net**, the statistics are obtained using the outer product operation from two different layers and pooled to obtain an utterance representation. The three architectures are shown in Fig. 1, with LID-net having been introduced in [19],. The main differences are that the spatial pyramid pooling (SPP) operation (which was also adapted from image processing [23]) is replaced by O2P from the same or different layers. The detailed theory and mechanism of bilinear pooling will be discussed in Section II-B while the proposed architectures are detailed in Section II-C. In Section III,we explore the strong modelling capability of the networks.

To summarise, the contribution of this paper is two novel end-to-end architectures named LID-net2 and LID-bilinear-net, that utilize LID-senones to obtain high-order statistics. Experiments on the full 23 languages of NIST LRE 2009 compare performance to state-of-the-art DBF/i-vector systems, demonstrating a very considerable improvement, especially for the shortest utterances.

II. Bilinear Models for LID

A. A Statistical View of LID-net

The structure of LID-net [19], shown in Fig.1(a), consists of a DNN-based front-end to derive LID-related acoustic features, followed by a CNN back-end, using SPP to form an utterance representation. The DNN is configured with a constricted bottleneck (BN) layer to transform acoustic features into a compact representation in a frame-by-frame manner. Convolutional layers then perform nonlinear transformations of BN features into units which are discriminative to language, termed LID-senones. The SPP layer forms an utterance representation from LID-senones, then the derived vector can be classified directly as described in [19].

The size[1] of LID-senone after convolutional *layer n* ($\boldsymbol{f_n}$) is $K_n@1 \times N_2$, and for convenience it can be reshaped to $K_n \times N_2$, then the LID-senone statistics ($\boldsymbol{N}$) are also reshaped from $K_n@1 \times 1$ to $K_n \times 1$. The $\boldsymbol{f_n}$ is transferred into $\boldsymbol{\gamma_n}$ after softmax $\boldsymbol{\gamma_n} = softmax(\boldsymbol{f_n})$. The elements of $\boldsymbol{\gamma_n}$ are $\gamma_{nk}(t)$ $(k = 1 \dots K_n$ and $t = 1 \dots N_2)$ while the elements of $\boldsymbol{N}$ are N_k $(k = 1 \dots K_n)$. Therefore if average pooling is used, zeroth order statistics are

$$N_k = \frac{1}{N_2} \sum_{t=1}^{N_2} \gamma_{nk}(t) \tag{1}$$

It is clear that with this method the kth senone statistic is computed just like the zeroth Baum-Welch statistic of acoustic features in the kth Gaussian in the standard i-vector system. The previous end-to-end system that used only zeroth order LID-senone statistics [19] outperformed state-of-the-art DBF/i-vector systems which utilized high-order statistics. Therefore utilizing higher order statistics obtained using the back-propagation algorithm in LID-bilinear-net would be expected to improve performance even further.

B. New Pooling Mechanism

The pooling model $\mathcal{B}$ in CNN can be viewed as $\boldsymbol{f_{A,B}} = \mathcal{B}(\boldsymbol{f_A}, \boldsymbol{f_B})$. Let $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$ be the A and B feature maps derived from structured CNN layers. In LID-net2, A and B are from the same layer feature maps, whereas in LID-bilinar-net, they are from different layer feature maps. In each case, $\boldsymbol{f_{A,B}}$ is the output of bilinear pooling. The size of $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$ are $(H \times W) \times K_A$ and $(H \times W) \times K_B$ respectively (reshaped from $K_A@H \times W$ and $K_B@H \times W$ respectively), implying both $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$ must have the same feature dimension W and H to be compatible, but could have different numbers of channels.

The expression of bilinear pooling can be developed to $\boldsymbol{f_{A,B}} = \mathcal{B}(\boldsymbol{f_A}, \boldsymbol{f_B}) = \mathcal{P}(\boldsymbol{f_A}^T \cdot \boldsymbol{f_B})$. The feature map outputs are combined at each location using the matrix outer product, thus the shape of $(\boldsymbol{f_A}^T \cdot \boldsymbol{f_B})$ is simply $K_A \times K_B$. To obtain an utterance representation descriptor, the pooling function $\mathcal{P}$ aggregates the bilinear feature across the entire spatial domain of one combination, and here we choose average pooling and so $\boldsymbol{f_{A,B}}$ will end up with size $K_A \times K_B$, effectively reshaped to $(K_A \times K_B)@1 \times 1$. The descriptor then can be used with a classifier, and here we use a multi-layer neural network.

C. Bilinear model for LID

Referring to the structure of the existing LID-net and proposed LID-bilinear-net shown in Fig.1, a DNN-based front-end extracts LID-features while a CNN-based back-end derives LID-senones. LID-bilinear-net's bilinear pooling layer extracts a high-order utterance representation utilizing correlation of dimensions in LID-senones. This utterance descriptor could then be directly used with a classifier, and the whole network can use back-propagation rather than typical high-order statistics algorithms such as FV [20] or O2P [21].

As Section II-A mentioned, feature maps $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$ could be reshaped into sizes of $K_A \times N_2$ and $K_B \times N_2$ respectively (where N_2 is the number of elements in each channel). Due to the filter size of *convolutional layer 1* covering the full LID-feature dimension, the height of feature maps after it are set to unity. Elements in feature map $\boldsymbol{f_A}$ are defined as $f_{Ad}(t)$ $(d = 1 \dots K_A$, $t = 1 \dots N_2)$ and in feature map $\boldsymbol{f_B}$ the element could be $f_{Bk}(t)$ $(k = 1 \dots K_B$, $t = 1 \dots N_2)$. After the softmax operation, $\boldsymbol{f_B}$ becomes $\boldsymbol{\gamma}$, which can be viewed as the posterior of corresponding LID-senones at frame level, with its elements defined as $\gamma_k(t)$ $(k = 1 \dots K_B$, $t = 1 \dots N_2)$. Following the mechanism of bilinear pooling, using the feature map $\boldsymbol{f_A}$ and its corresponding posterior $\boldsymbol{\gamma}$, the bilinear pooling models the first order LID-senone statistics,

$$\boldsymbol{f_{AB}(k)} = \frac{1}{N_2} \sum_{t=1}^{N_2} \gamma_k(t) \cdot \boldsymbol{f_A(t)} \tag{2}$$

With feature maps $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$, the bilinear pooling can also model the second order LID-senone statistics with vectorization expression

$$\boldsymbol{f_{AB}} = \frac{1}{N_2} \boldsymbol{f_A}^T \cdot \boldsymbol{f_B} \tag{3}$$

If $\boldsymbol{f_A}$ and $\boldsymbol{f_B}$ come from the same layer in the CNN, this would be the standard formula to calculate O2P (e.g. eqn.(2) in [21]).

The high-order LID-senone statistics can not only cover a wide speech context, but also extract the relationship along its feature dimension. Typically, i-vector methods do not learn the feature extractor functions, with only the parameters of the encoder being learnt. Furthermore, even though an i-vector is compact, its training procedure is not end-to-end. The advantages of LID-net2 and LID-bilinear-net are to learn the feature extractor and encoder simultaneously, allowing the whole network to be easily fine-tuned. Owing to the flexibility of CNNs, the input feature maps of bilinear pooling can be either from the same or different layers. We believe that bilinear pooling from different input layers can further improve performance since the information that they contain is to some extent complementary.

D. Training Procedure

Due to the large quantity of training parameters, many of which are in the *full connection layers*, and the fact that LID-net, LID-net2 and LID-bilinear-net share a structure for their front end, we initialize the network with the trained LID-net parameters, then train the two new networks from this. The process is namely: (1)

1) Train a 6 layer DNN (48×21-2048-2048-50-2048-2048-3020) with an internal bottleneck layer using a large scale ASR corpus (SwitchBoard);
2) Transfer parameters from the first 3 layers to *DNN layer1-layer3* of LID-net and train LID-net;
3) Transfer all layer parameters below the SPP layer to LID-net2 and LID-bilinear-net and train the two new networks separately.

Steps (1) and (2) are the same as for LID-net so detailed information can be found in [19]. Step (3) is described below.

[1] A size of $K_n@1 \times N_2$ means the height is 1, the number of weights is N_2 and there are K_n channels.

III. Experimental evaluation

A. Experiments Setup

To evaluate the effectiveness of the proposed network, we conduct extensive experiments with the NIST LRE09 corpus comprising 23 languages. Equal error rate (EER) and C_{avg} are used to measure performance. Due to the evaluations being performed on 30s, 10s and 3s temporal scales, when training the two shorter scales, we randomly extract shorter speech segments from the 30s training dataset during each epoch. For comparison, the following system are implemented.

DBF/i-vector: This is the state-of-the-art baseline system used for comparison. The i-vector method uses DBF as front-end features and back-end modeling from a well-trained DNN trained on ASR data. LDA and WCCN compensate the variability, and cosine distance is used to obtain the final score.

LID-net: The end-to-end network in [19] is used for comparison. This only employs zeroth order Baum-Welch statistics from LID-senones.

LID-net2: The first new network proposed in this paper, where high-order statistics of LID-senones are obtained through second order pooling (O2P) of posteriors pooled from the CNN convolution layer prior to the fc mapping network.

LID-bilinear-net: As above but instead of using outputs from a single convolution layer, this network utilizes posteriors pooled from two different CNN layers.

Each network is trained and tested independently for 30s, 10s and 3s duration data. For LID-net and variants, cosine distances on corresponding language posteriors are directly utilized to obtain scores without LDA and WCCN.

B. Configuration of new networks

Three separate copies of each system are trained for the different time scales (3s, 10s and 30s). Each have 6 convolutional layers. The feature maps from CNN layers 1-5 have 512 channels and the feature maps after layer 6 are evaluated with between 64 and 512 channels. Each convolutional layer is followed by a batch normalization layer [24] and first and second order LID-senone statistics are evaluated. The feature map $\boldsymbol{f}$ is obtained before the batch normalization while the feature map γ is extracted from a convolutional layer output followed by a softmax operation. The input of the LID-senone pooling process could be from either the same or different feature maps. In LID-net2, these are obtained from after convolutional 6; whereas LID-bilinear-net performs bilinear pooling with input feature maps from convolutional layers 5 and 6.

C. Experiments on LID-net and DBF/i-vector

Before training the new networks, we must train the corresponding LID-net first. This also has six convolutional layers, and must also be trained with 64 to 512 channels in the feature map after layer 6 for comparison. The performance of various LID-net configurations is shown in Table I alongside the current state-of-the-art DBF/i-vector system. The notation LID-net-64 means the feature map after CNN layer 6 has 64 channels.

TABLE I
Comparison of performance between LID-net and DBF/i-vector in EER (%) for all systems and scales.

EER	3s	10s	30s
DBF/i-vector	10.79	3.05	1.48
LID-net-64	7.76	2.92	1.54
LID-net-128	7.58	2.89	1.55
LID-net-256	**7.57**	**2.66**	**1.46**
LID-net-512	7.79	2.81	1.50

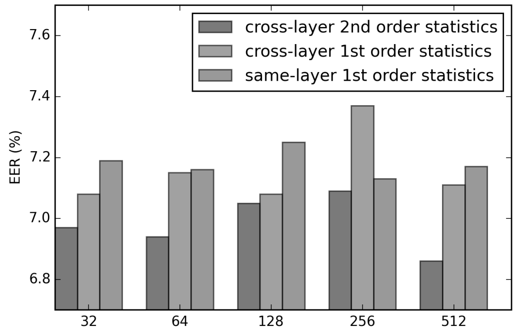

Fig. 2. Histogram of 3s EER performance for same-layer pooling (LID-net) and cross-layer pooling (LID-bilinear-net) incorporating both first and second order statistics.

Thanks to the end-to-end nature of LID-net, it achieves better performance than the baseline DBF/i-vector system over all scales. In general, the shorter the segment, the greater the advantage for LID-net. The compelling improvement achieved by LID-net at almost all scales lends confidence to the ability of the discriminative training procedure. As far as we concerned, the discriminative model can handle the variance of speakers, channels and noise in short utterances better than a generative model. However the number of channels should not be too small or too large, as too many trained parameters leads to over-fitting whereas too few parameters cannot model the LID-senones effectively.

D. Evaluation on LID-net2 LID-bilinear-net

After transferring trained LID-net parameters to the corresponding LID-net2 and LID-bilinear-net systems, we re-train using the same training data, and verify whether higher order statistics from the pooling improves performance. Focusing only on the most difficult 3s utterances, we conducted extensive experiments to explore the mechanism for computing the statistics through same- or cross-layer pooling.

Fig. 2 shows EER performance for various systems and feature dimensions on 3s utterances. Results are shown for both LID-net2 and LID-bilinar-net, with the latter computed using either first or second order statistics. Comparing with Table I we first see that both systems outperform LID-net (EER=7.57 for 3s utterances), demonstrating clearly the robustness that is gained by using higher-order LID-senone statistics. Cross-layer bilinear pooling performs better than same-layer pooling, and we argue that computing statistics across layers provides some degree of complementary information as well as perhaps some improvement in robustness. This is demonstrated by LID-bilinear-net outperforming LID-net2 in every condition. We thus evaluate LID-bilinear-net more fully.

TABLE II
COMPARISON OF PERFORMANCE OF CROSS LAYER LID-BILINEAR-NET FOR ALL SCALES IN EER (%) WITH DIFFERENT DIMENSIONS.

EER	3s	10s	30s
DBF/i-vector	10.79	3.05	1.48
64-relu	6.94	2.40	**1.48**
128-relu	7.05	2.33	1.59
256-relu	7.09	**2.32**	1.58
512-relu	**6.86**	2.43	1.51

Table II includes 3s, 10s and 30s LID-bilinear-net results, for different numbers of channels in the output layer. Performance is good compared to Table I, although the 30s result seems to be data-limited rather than architecture-limited (LID-bilinear-net and LID-net2 have more parameters to train than LID-net through having an additional fully connected output layer). Note that the bilinear pooling method demonstrates its compactness: just 64 channels in LID-bilinear-net outperforms both the DBF/i-vector and the LID-net systems for shorter utterances in terms of EER.

IV. CONCLUSION

This paper has introduced two novel end-to-end neural networks, named LID-net2 and LID-bilinear-net. Both systems share their trained lower DNN layer parameters with LID-net, a previous DNN/CNN network that did not incorporate bilinear pooling and could only utilize zeroth order statistics. In all systems, DNN layers are first used to extract LID-features from acoustic training features, then LID-senones obtained by CNN through several convolutional layers which span a time context. LID-senones are thought to be discriminative to languages in the way that senones are discriminative to phonetic content. The LID-senone derivation is followed by a pooling layer that spans from frame to utterance level, from which high-order (first and second order) statistics are computed. In LID-net2 these are dervied from convolutional layer 6 whereas in LID-bilinear-net they are computed cross-layer from feature maps of convolutional layers 5 and 6. Each system is trained end-to-end via back-propagation with LID labels. The performance of each LID-net variant is better than state-of-the-art DNN/i-vector systems, and the two novel networks in this paper perform best of all, with LID-bilinear-net demonstrating the highest performance and greatest degree of robustness.

REFERENCES

[1] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction." *Proc. of Interspeech*, pp. 857–860, 2011.

[3] Y. Song, X. Hong, B. Jiang, R. Cui, I. V. McLoughlin, and L. Dai, "Deep bottleneck network based i-vector representation for language identification," *Proc. of InterSpeech*, pp. 398–402, 2015.

[4] F. Richardson, D. Reynolds, and N. Dehak, "A unified deep neural network for speaker and language recognition," *arXiv preprint arXiv:1504.00923*, 2015.

[5] B. Jiang, Y. Song, S. Wei, J.-H. Liu, I. V. McLoughlin, and L.-R. Dai, "Deep bottleneck features for spoken language identification," *PLoS ONE*, vol. 9, no. 7, 2014.

[6] Y. Song, B. Jiang, Y. Bao, S. Wei, and L.-R. Dai, "I-vector representation based on bottleneck features for language identification," *Electronics Letters*, vol. 49, no. 24, pp. 1569–1570, 2013.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[9] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Computing Surveys (Csur)*, vol. 40, no. 2, p. 5, 2008.

[10] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks." *International Conference on Machine Learning*, vol. 14, pp. 1764–1772, 2014.

[11] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," *Proc. of ICASSP*, pp. 5337–5341, 2014.

[12] D. Garcia-Romero and A. McCree, "Stacked long-term TDNN for spoken language recognition," *Proc. of Interspeech*, pp. 3226–3230, 2016.

[13] G. Gelly, J.-L. Gauvain, V. Le, and A. Messaoudi, "A divide-and-conquer approach for language identification based on recurrent neural networks," *Proc. of Interspeech*, pp. 3231–3235, 2016.

[14] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno, "Automatic language identification using long short-term memory recurrent neural networks," *Proc. InterSpeech*, 2014.

[15] W. Geng, W. Wang, Y. Zhao, X. Cai, and B. Xu, "End-to-end language identification using attention-based recurrent neural networks," *Proc. of Interspeech*, pp. 2944–2948, 2016.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[17] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[19] M. Jin, Y. Song, I. McLoughlin, L.-R. Dai, and Z.-F. Ye, "LID-senone extraction via deep neural networks for end-to-end language identification," *Proc. of Odyssey*, 2016.

[20] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," *European conference on computer vision*, pp. 143–156, 2010.

[21] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Semantic segmentation with second-order pooling," *European Conference on Computer Vision*, pp. 430–443, 2012.

[22] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *346–361*, 2014.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.