INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

End-to-End Language Identification Using High-Order Utterance Representation with Bilinear Pooling

Ma Jin[1], *Yan Song* [1], *Ian McLoughlin*[2], *Wu Guo* [1], *Li-Rong Dai*[1]

[1]National Engineering Laboratory of Speech and Language Information Processing
University of Science and Technology of China, Hefei, China
[2]School of Computing, University of Kent, Medway, UK

jinma525@mail.ustc.edu.cn, {songy, lrdai, guowu}@ustc.edu.cn, ivm@kent.ac.uk

Abstract

A key problem in spoken language identification (LID) is how to design effective representations which are specific to language information. Recent advances in deep neural networks have led to significant improvements in results, with deep end-to-end methods proving effective. This paper proposes a novel network which aims to model an effective representation for high (first and second)-order statistics of LID-senones, defined as being LID analogues of senones in speech recognition. The high-order information extracted through bilinear pooling is robust to speakers, channels and background noise. Evaluation with NIST LRE 2009 shows improved performance compared to current state-of-the-art DBF/i-vector systems, achieving over 33% and 20% relative equal error rate (EER) improvement for 3s and 10s utterances and over 40% relative C_{avg} improvement for all durations.

Index Terms: language identification, utterance representation extraction, end-to-end neural network, bilinear pooling

1. Introduction

The key problem for language identification (LID) is how to distill an efficient and compact representation specific to LID information. This is challenging due to large variation in speech content, speakers, channels and background noise, coupled with a scarcity or mismatch in training resources. At present, total variability (TV) methods 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, 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 achieved impressive results in applications like 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. For example, Song *et.al*, Richardson *et.al* and Jiang *et.al* [3, 4, 5] proposed using deep bottleneck features (DBFs) from a well trained DNN for automatic speech recognition (ASR) [6]. DBFs are inherently robust to phonotactically irrelevant information. Lei *et.al*, Kenny *et.al* and Ferrer *et.al* [7, 8, 9] proposed collecting sufficient statistics using a structured DNN to form effective representations from posteriors of phoneme or phoneme states. DNNs have been shown to excel when combined with phonotactic training in LID modelling, nevertheless both the DBFs and calculated statistics are extracted from phoneme or phoneme states, which are not always discriminative to languages.

To extract language discriminant features and representations, more and more end-to-end NNs have been proposed to span frame level to utterance level LID identity – avoiding the need for discriminative back-end algorithms. End-to-end schemes have been used in image processing [10, 11, 12] and speech recognition [13], combining good performance with convenience in training.

Lopez-Moreno *et.al* [14] proposed an end-to-end scheme for LID using large scale DNNs, which performed well. 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* [15] improved this by introducing a time delay neural network (TDNN), which spans a wider temporal context. A bottom-up hierarchical structure used to produce a posterior probability over the set of languages concatenated over a long time span. Gelly *et.al* [16] and Gonzalez *et.al* [17] 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* [18] applied attention-based RNN mechanisms, first used in neural machine translation, to LID. Each speech frame has a posterior, forming vectors that are weighted and summed into one utterance representation. This unified architecture allowed end-to-end training, and boosted system performance.

Compared to LSTM-RNN, convolutional neural networks (CNN) have more flexibility with many variant architectures [19, 20, 21]. In our previous work [22], a novel end-to-end approach named **LID-net** was proposed, combining the proven frame-level feature extraction capabilities of the DNN with the effective utterance level mapping abilities of the CNN. This allowed language discriminant features to be obtained, which we termed **LID-senones**. Performance was good compared to state-of-the-art DBF/i-vector systems, particularly for short utterances, however LID-net only averaged LID-senone posteriors using zeroth order Baum-Welch statistics.

The above end-to-end networks have demonstrated the capability of discriminative modelling. However instead of modelling an utterance as LID-senones in the time dimension, the bilinear pooling computes the output product of LID-senone sequences from two CNN layers. This yields an utterance repre-

http://dx.doi.org/10.21437/Interspeech.2017-44

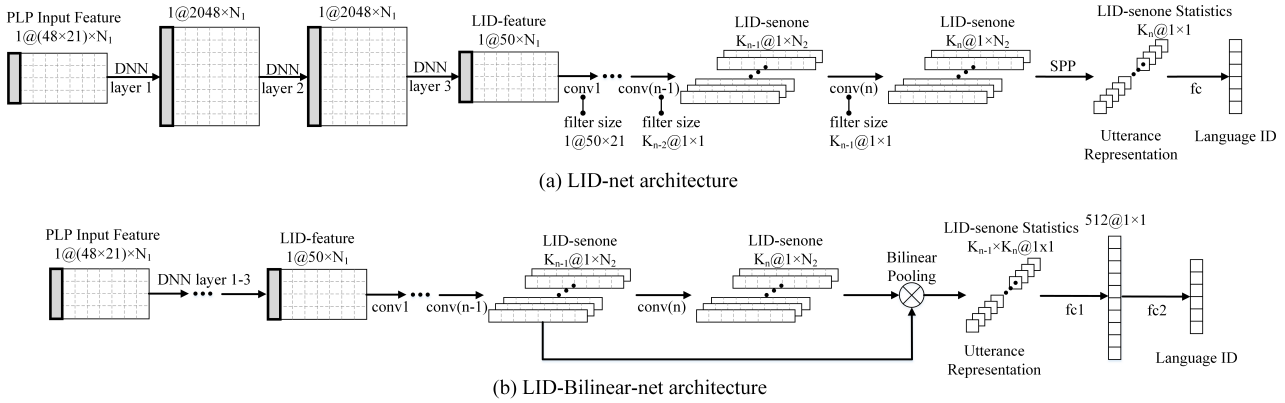

Figure 1: ***LID-net*** *(top) where features are extracted frame-by-frame from DNN layers 1-3. LID-senones are obtained through several convolutional layers, with the expansion of filter size in convolutional layer* 1 *to a context of 21 frames, followed by several* 1×1 *filters (convolutional layers* 2 *to* n*).* ***LID-bilinear-net*** *(bottom) is identical to LID-net up to the bilinear pooling layer. This is the outer product of two feature maps from lower convolutional layers, from which first and second order statistics can be obtained.*

sentation in terms of LID-senone statistics that is invariant to the time dimension of the original recording, and is considered to be more robust to within-class variance, channels and background noise. The output representation acts like a covariance matrix formed between the same or two different layers of LID-senones, from which LID statistics are obtained.

This approach 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) [23] and second order pooling (O2P) [24] respectively, showing that first and second order statistics, widely used in patten recognition, can contribute outstanding performance to classification.

1.1. Contribution

We introduce an end-to-end DNN-CNN neural network that utilizes high-order LID-senone statistics. This system, named **LID-bilinear-net**, combines the advantage of both the high-order Baum-Welch statistics calculation of i-vector systems and the natural discriminant attributes of neural networks. High-order statistics are obtained through a *bilinear pooling model* borrowed from fine-grained visual recognition [25]. Two convolutional layer outputs are combined using outer product multiplication at each dimension of the LID-senone and pooled to obtain an utterance representation. The architecture of LID-bilinear-net, shown in Fig. 1, is based upon that of LID-net [22], except the bilinear pooling layer replaces the original single-layer spatial pyramid pooling (SPP) (which was also adapted from image processing [26]). First and second order statistics can then be obtained from the bilinear pooling. To summarise, the contribution of this paper is a novel end-to-end architecture named LID-bilinear-net, that utilizes 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.

In the remainder of this paper, the detailed theory and mechanism of bilinear pooling will be discussed in Section 2.2 while the proposed LID-bilinear-net architecture is detailed in Section 2.3. In Section 3, the task is outlined before extensive experiments to explore the strong modelling capability of LID-bilinear-net. Section 4 will conclude the paper.

2. Bilinear Models for LID

2.1. A Statistical View of LID-net

The structure of LID-net [22], 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 [22].

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

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 [22] 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.

2.2. Bilinear Pooling Mechanism

The formation of a bilinear 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; A and B could be from the same or different layer feature maps. $\boldsymbol{f_{A,B}}$ 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

[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.

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.

2.3. 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 [23] or O2P [24].

As Section 2.1 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,

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

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

$$\boldsymbol{f_{AB}} = \frac{1}{N_2} \boldsymbol{f_A}^T \cdot \boldsymbol{f_B} \tag{2}$$

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 [24]).

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 advantage of LID-bilinear-net is 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.

2.4. Training Procedure

Due to the large quantity of training parameters in LID-bilinear-net, many of which are in the *full connection layer*, and the fact that LID-net and LID-bilinear-net share a structure for their first half, we initialize the network with the trained LID-net parameters, then train the new network directly. The process is namely:

(1) Train a 6 layer DNN (48×21-1024-1024-50-1024-1024-3020) with an internal BN layer using 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-bilinear-net and train LID-bilinear-net.

Steps (1) and (2) are the same as for LID-net so detailed information can be found in [22]. Step (3) is described below.

3. Experimental evaluation

3.1. 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 crop short speech segments from the recordings that make up the 30s training dataset. 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 [22] is used for comparison. This only employs zeroth order Baum-Welch statistics from LID-senones.

LID-bilinear-net: The new network proposed in this paper, where high-order statistics of LID-senones can be obtained via the end-to-end scheme utilizing 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 LID-bilinear-net, cosine distances on corresponding language posteriors are directly utilized to obtain scores without LDA and WCCN.

3.2. Configuration of LID-bilinear-net

Separate LID-bilinear-net systems for different scales are trained with 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 32 and 512 channels. Each convolutional layer is followed by a batch normalization layer [27] 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 $\boldsymbol{\gamma}$ is extracted from a convolutional layer output followed by a softmax operation. The input of the bilinear pooling process could be from either the same or different feature maps, so two configurations of bilinear pooling input are evaluated: one is same-layer bilinear pooling with input feature maps from after convolutional 6; the other is cross-layer bilinear pooling with input feature maps from convolutional layers 5 and 6.

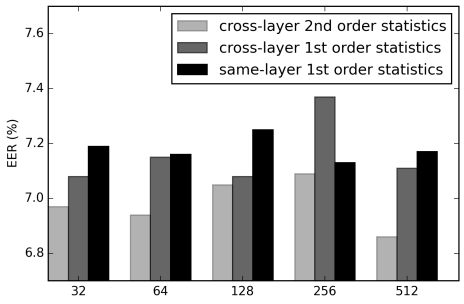

Figure 2: *Evaluation of LID-bilinear-net on 3s utterances. Results are shown in EER (%), for same-layer pooling and cross-layer pooling of first and second order statistics.*

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

Before training LID-bilinear-net, we must train the corresponding LID-net first. This also has six convolutional layers, and must also be trained with 32 to 512 channels in the feature map after layer 6 for comparison. The performance of various LID-net configurations is shown in Table 1 alongside the current state-of-the-art DBF/i-vector system. The notation LID-net-32 means the feature map after CNN layer 6 has 32 channels.

Table 1: *Comparison between LID-net and DBF/i-vector. Performance is given in EER (%) and C_{avg} (%) for all systems and scales.*

System	3s		10s		30s	
	EER	C_{avg}	EER	C_{avg}	EER	C_{avg}
DBF/i-vector	10.79	7.48	3.05	2.14	1.48	1.10
LID-net-32	7.67	6.02	2.74	1.54	1.49	1.05
LID-net-64	7.76	5.99	2.92	1.64	1.54	0.75
LID-net-128	7.58	6.15	2.89	2.00	1.55	0.91
LID-net-256	**7.57**	**5.05**	**2.66**	**1.46**	**1.46**	1.21
LID-net-512	7.79	6.64	2.81	1.49	1.50	**0.74**

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.

3.4. Evaluation on LID-bilinear-net

After transferring trained LID-net parameters to the corresponding LID-bilinear-net, we re-train using the same training data, and verify whether bilinear pooling improves performance further. Focusing on 3s utterances, we conduct extensive experiments to explore the mechanism for computing first/second order statistics through same- or cross-layer pooling.

Fig. 2 shows EER performance for various systems on 3s utterances. The number N along the x axis indicates that the LID-bilinar-net system was initialised from LID-net-N. Results are shown for both same-layer pooling and cross-layer pooling, with the latter computed using either first or second order statistics. Comparing with Table 1 we first see that all LID-bilinear-net systems outperform LID-net. This is thanks to the robustness that is gained by using high-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. Results also show that using the second order statistics is more robust in every case than that from first order statistics. Therefore the following evaluations only list the performance of second order statistics of LID-senones obtained from cross-layer bilinear pooling.

Table 2: *Evaluations on cross layer LID-bilinear-net for all scales. Performance is given in EER (%) and C_{avg} (%) for all test conditions.*

LID-bilinear-net	3s		10s		30s	
	EER	C_{avg}	EER	C_{avg}	EER	C_{avg}
32-relu	6.97	6.20	2.39	**1.20**	1.52	0.77
64-relu	6.94	5.32	2.40	1.38	**1.48**	0.95
128-relu	7.05	5.52	2.33	1.50	1.59	**0.66**
256-relu	7.09	5.26	**2.32**	1.74	1.58	0.87
512-relu	**6.86**	**4.38**	2.43	1.46	1.51	0.87

Table 2 includes 3s, 10s and 30s LID-bilinear-net results, for different numbers of channels in the output layer. Performance is good compared to Table 1, although the 30s result seems to be data-limited rather than architecture-limited (LID-bilinear-net has 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.

4. Conclusion

This paper has introduced a novel end-to-end neural network, named LID-bilinear-net. DNN layers are first used to extract LID-features from acoustic training features, then LID-senones obtained 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 bilinear pooling layer that spans from frame to utterance level, from which high-order (first and second order) statistics are computed. The system is trained end-to-end via back-propagation. LID-bilinear-net shares lower layer trained parameters with LID-net, a previous DNN/CNN network that did not incorporate bilinear pooling and could utilize only zeroth order statistics. Experimental results demonstrate the strong modelling capability of LID-bilinear-net, achieving relative improvements in EER of over 33% and 20% for 3s and 10s durations and over 40% relative improvement in C_{avg} for all durations, compared to the current state-of-the-art DBF/i-vector system.

5. Acknowledgements

The authors would like to acknowledge the support of National Natural Science Foundation of China grant no U1613211.

6. 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] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," *Proc. of ICASSP*, pp. 1695–1699, 2014.

[8] P. Kenny, V. Gupta, T. stafylakis, P. Quellet, and J. Alam, "Deep neural networks for extracting Baum-Welch statistics for speaker recognition," *Proc. of ICASSP*, pp. 1695–1699, 2014.

[9] L. Ferrer, Y. Lei, M. McLaren, and N. Scheffer, "Study of senone-based deep neural network approaches for spoken language recognition," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 24, no. 1, pp. 105–116, 2016.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

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

[25] 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.

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

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