Hypersphere Embedding and Additive Margin for Query-by-example Keyword Spotting

Haoxin Ma*†, Ye Bai*†, Jiangyan Yi* and Jianhua Tao*†‡
* National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, China
† School of Artificial Intelligence, University of Chinese Academy of Sciences, China
‡ CAS Center for Excellence in Brain Science and Intelligence Technology, Institute of Automation, Chinese Academy of Sciences, China
E-mail: mahaoxin2019@ia.ac.cn, {ye.bai, jiangyan.yi, jhtao }@nlpr.ia.ac.cn

***Abstract*—Query-by-example (QbE) keyword spotting is convenient for users to define their own keywords, so it is useful in device control. However, conventional regular softmax, which is commonly used for training QbE models, has two limitations. First, the learned features are not discriminative enough. Second, norm variations of the unnormalized features affect computing cosine similarities. To address these issues, this paper introduces normalization and additive margin into residual networks for QbE keyword spotting. Features and weights are normalized on a hypersphere of fixed radius. Additive margin further helps to reduce the intra-class variations and increase inter-class differences. Based on public datasets AISHELL-1 and HelloNPU, we design three different test sets, namely in-vocabulary, out-of-vocabulary, and cross-corpus, to evaluate our proposed method. Experiments show that our proposed method can learn more discriminative embedding features. For totally unseen situation, our proposed method achieves a relative false rejection rate reduction of 46.60% when the false alarm rate is 2% in cross-corpus evaluation, compared with regular softmax.**

I. Introduction

Query-by-example (QbE) keyword spotting is the task of detecting the predefined keyword in a series of speech recordings. The most typical application is to activate a device by a wake-up word. QbE is helpful especially when users want to define their own special word rather than using the pre-set phrase. QbE keyword spotting system detects audio segments directly without the need to build a robust automatic speech recognition system. Besides, it can easily deal with the out-of-vocabulary (OOV) and low-resource situations.

Previous studies on QbE methods can be divided into two categories. One is the dynamic time warping (DTW) based methods [1], [2], [3] which calculate similarity of the frame-level feature sequences. However, DTW costs much time and computation. The other category is the embedding learning methods which project the acoustic features into a fixed-dimensional space and evaluate the similarity of embedding vectors, which is simple but efficient. Several works [4], [5], [6], [7] demonstrate the success of embedding learning and it outperforms the DTW in QbE keyword spotting. Besides, many neural network architectures, such as convolutional neural networks (CNN) [8], long short-term memory (LSTM) networks [9], attention-based networks [10], are used as embedding extractor. Usually, regular softmax, namely cross-entropy is used for training [5], [7], [10]. Features learned from the networks with regular softmax loss have an intrinsic angular distribution with different norms [11], so cosine similarity is effectively used for testing. To predict whether the speech contains the query term, the distance between the embeddings of the same words should be close and vice versa. However, regular softmax loss can only separate different classes away without making features of the same class compact, which leads to a limitation in discriminative feature learning [11], [12]. Besides, there is a gap left between training and testing, because the feature representation in training is irrelevant to the evaluation criterion (cosine similarity) in testing. Yuan et al. use triplet loss to reduce intra-class distance and increase interclass distance [13]. However, it is difficult to mine hard triplets and train networks [14], [15], [16]. Moreover, improper hard negative mining can lead to collapsed models [17].

Normalization is a helpful method to eliminate the gap between training and testing. Feature normalization can boost the performance in facial recognition [18], [19]. Additionally, additive margin softmax (AM-softmax) [12] is proposed to improve the softmax loss and works well in facial verification recently. It introduces an additive margin via subtracting a hyper-parameter m in the cosine space [20]. It can minimize the intra-class variation and avoid the hard triplets mining process at the same time.

This paper introduces normalization and AM-softmax into QbE keyword spotting to extract the embedding features. Both weights and features are normalized on a hypersphere. Thus, the inner product of them is the same as cosine similarity. Moreover, intra-class distances are reduced by AM-softmax. This paper also incorporates the residual networks (ResNet) motivated by its success in image classification [21], [22] and speaker verification [23]. Cosine similarity is used to evaluate the similarity. Our experiments are conducted on a modified AISHELL-1 dataset [24] and HelloNPU corpus [25]. We design three test sets to test the effectiveness in different situations. They are in-vocabulary set, OOV set, and cross-corpus set. The results illustrate that normalization with AM-softmax can improve the performance compared to regular softmax. At false alarm rate (FAR) of 2%, the false rejection rate (FRR) is relatively reduced by 79.86% on in-vocabulary

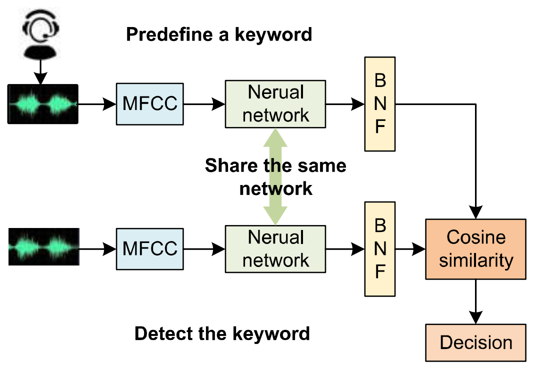


Fig. 1. The framework for QbE keyword spotting.

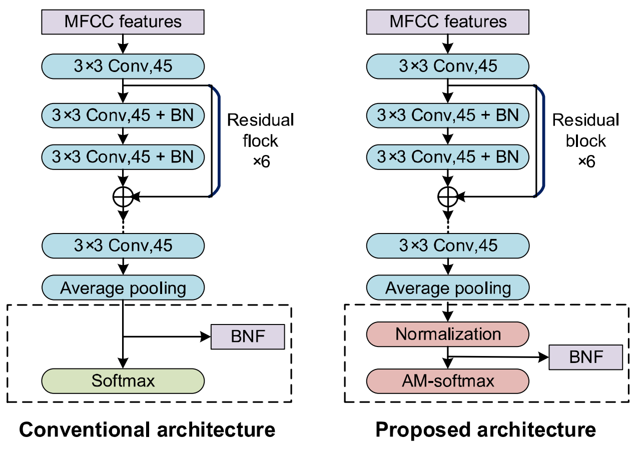


Fig. 2. The left conventional architecture is trained with regular softmax loss, and the right one introduces normalization and AM-softmax loss. The blue parts are the networks: residual block $\times$ 6 means that six residual blocks are stacked. The parts with dashed line show the training methods.

set, 68.03% on OOV set, and 46.60% on cross-corpus set, respectively.

In the rest of the paper, the framework is described in section 2. The proposed methods are described in section 3. Section 4 illustrates the experimental results. Section 5 presents the conclusions and future work.

II. QbE Framework

As shown in Figure 1, in practice, a user can predefine a specific keyword, and its embedding feature is extracted by the neural network from the user's speech recordings. We name the embedding feature as bottleneck feature (BNF), the "BNF" in the figure, because it is generated from a layer's output. When an audio is inputted, the BNF of it comes out from the same neural network. Then, the system calculates the cosine similarity to make a decision whether the audio is the predefined keyword.

During training, audio segments x_{train} with labels l_{train} are used to train a classifier. The structure before the softmax layer projects the input data to the fixed-dimensional embedding features. The softmax layer matches the embedding features to each label to train the neural network. As demonstrated on the left of contrast, for conventional architecture, the loss function is regular softmax loss. The softmax is removed at test stage.

The regular softmax loss is as follows:

$$\begin{aligned} L_s &= -\frac{1}{n}\sum_{i=1}^{n} log\frac{e^{w_{y_i}^T y_i}}{\sum_{j=1}^{k} e^{w_{y_j}^T y_i}} \\ &= -\frac{1}{n}\sum_{i=1}^{n} \log\frac{e^{||w_{y_i}||||y_i||\cos\theta_{y_i}}}{\sum_{j=1}^{k} e^{||w_j||||y_i||\cos\theta_j}}, \end{aligned} \tag{1}$$

where y is the input of the softmax layer and is used as BNF. Vector w is the weight which stands for the prototype of the class. Subscripts i and j denote the i-th or the j-th sample. k and n donote the sizes of classes and samples, respectively. One class corresponds to one output label. According to vector inner product formula, we expand the formula out and get θ as the intersection angle. Thus, the softmax operation can be viewed as a similarity evaluation between samples and labels to help train the network.

When it comes to testing, the system calculates cosine similarity of the BNF for every audio pair. Then it comes out a "same or different" prediction based on a particular threshold. If the similarity score is over the threshold, the keyword is detected.

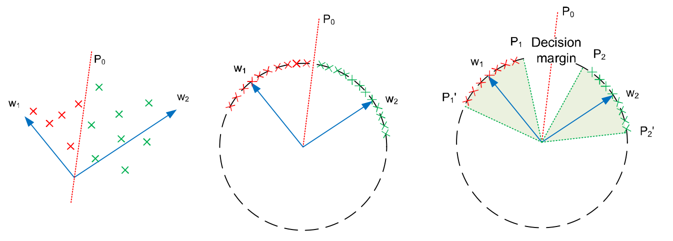
III. Hypersphere Embedding and Additive Margin Methods

Our goal is to verify the keyword from large amount of speech. But the embedding features learned by regular softmax loss are not discriminative enough. Besides, the various norms of the embedding features influence the performance of keyword spotting. Thus, normalization and AM-softmax are employed in ResNet. contrast shows our methods in comparison to the conventional training method.

A. Normalizatin operation

Hypersphere embedding is done by normalization operation, simply but effectively. As shown in Figure 3(b), both the feature vectors y (the "$\times$" marks in Figure 3(b)) and the weight vectors w are normalized on a unit hypersphere. For convenience, we just take a unit circle and two-dimensional problem as example. We name the weight vectors as the prototypes of each classes. Compared with the unnormalized one in Figure 3(a), radial variations of the vectors are removed. Thus, the calculation can just focus on the angular similarity. The smaller intersection angle between the feature vectors and prototypes, the higher possibility they are the same classes. If without normalization, there will be a confusion that the vectors with small norms have huge variation in angle although their Euclidean distances are small. This confusion can result in higher errors in testing under the cosine metric [19].

Hypersphere embedding avoids radial variations. Besides, after normalization, the inner product operation wy of softmax becomes the cosine calculation, compatible with the cosine similarity.



(a) Regular (b) Normalization (c) AM-softmax

Fig. 3. A comparison of regular softmax, normalization operation, and AM-softmax: the "$\times$" marks represent the embedding vectors of samples. Different colors correspond to different labels.

B. AM-softmax loss function

In addition to eliminating radial variations, the cosine similarity scores between embeddings of the same classes should be as high as possible, and the cosine similarity scores between embeddings of different classes should be as low as possible. Then, larger margin will be left for judgment. This can be done by AM-softmax.

In AM-softmax loss, a cosine margin m is introduced. The function of AM-softmax is as follows:

$$\begin{aligned} L_{AM-s} &= -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{s(\cos\theta_{y_i}-m)}}{e^{s(\cos\theta_{y_i}-m)}+\sum_{j=1,j\neq i}^{k}e^{s\cos\theta_i}} \\ &= -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{w_{y_i}^{T}y_i}}{e^{s(w_{y_i}^{T}y_i-m)}+\sum_{j=1,j\neq i}^{k}e^{sw_j^{T}y_i}}. \end{aligned} \quad (2)$$

All the symbols stand for the same meaning as in section 2. m is the additive margin. s is the scaling factor. Since the vectors y and w are normalized, $wy = \cos\theta$. Cosine values can be calculated directly. The benefits of the margin subtraction are obvious. Because an additive margin m is subtracted from $\cos\theta$, the value of AM-softmax is less than the corresponding regular softmax one. If the value of $\cos\theta - m$ wants to be the same as the regular softmax, a larger cosine value is needed. Thereby the distance between the sample of the same label will be more compact and the intra-class differences will be reduced. Besides, a hyper-parameter s is used to scale the cosine values to yield better-performing features [18], [26]. AM-softmax loss can converge easier with proper scaling factor s.

Actually, the decision boundary is optimized in the cosine space rather than the angular space [20]. But the boundary in the cosine space is the same as the boundary in the angular space because there is a one-to-one mapping between θ and $\cos\theta$ ($0 < \theta < \pi$). To describe the AM-softmax function more simply and intuitively, the cosine margin is shown in the angular space in Figure 3(c).

We denote w_i as the prototype of corresponding class and P_i as the decision boundary. Taking two-dimensional problem as examples, Figure 3(a) is a schematic of regular softmax. The decision boundary of two classes is P_0 and the norms of two prototypes are different. As for AM-softmax in Figure 3(c), the decision boundary becomes a decision margin rather than one simple vector boundary P_0 [12]. The decision margin between class 1 for class 1 is from P_1 to P_2. From the formula$_a m, we have \cos\theta_{w_2P_1} - m = w_2P_1 - m = w_1P_1$. Thus, $m = w_1P_1 - w_2P_1 = w_2P_2 - w_1P_2$. The differences between the cosine scores for the adjacent decision boundaries should be the margin m. The larger the margin, the larger the difference.

IV. Experiments

A. Datasets

Since there is few publicly available dataset used for QbE keyword spotting, we develop our new dataset based on the existing AISHELL-1 dataset [24] and HelloNPU corpus [25]. According to the data preparation idea in the work of A. Jansen *et al.* [27], we select speech segments from the forced alignments of transcripts. The duration of the segments are at least 0.5 seconds and not exceeding 1 second. The labels contains at least 2 characters as texts. We choose the segments with the frequency of the top 5,000. Then we divide them into disjointed sets named as training set and development set, including 200,095 utterances and 25,604 utterances, respectively.

For evaluation, we design 3 types of sets including in-vocabulary set, OOV test set, and cross-corpus test set. Since the cross-corpus set is a totally different dataset whose data source and related field are completely irrelevant to AISHELL-1 dataset, we can have a further evaluation of our methods. Each test size is made up of 20,000 speech segment pairs. Half of them are positive and half of them are negative, which means half of them have the same labels and half of them have different labels. The details are as follows:

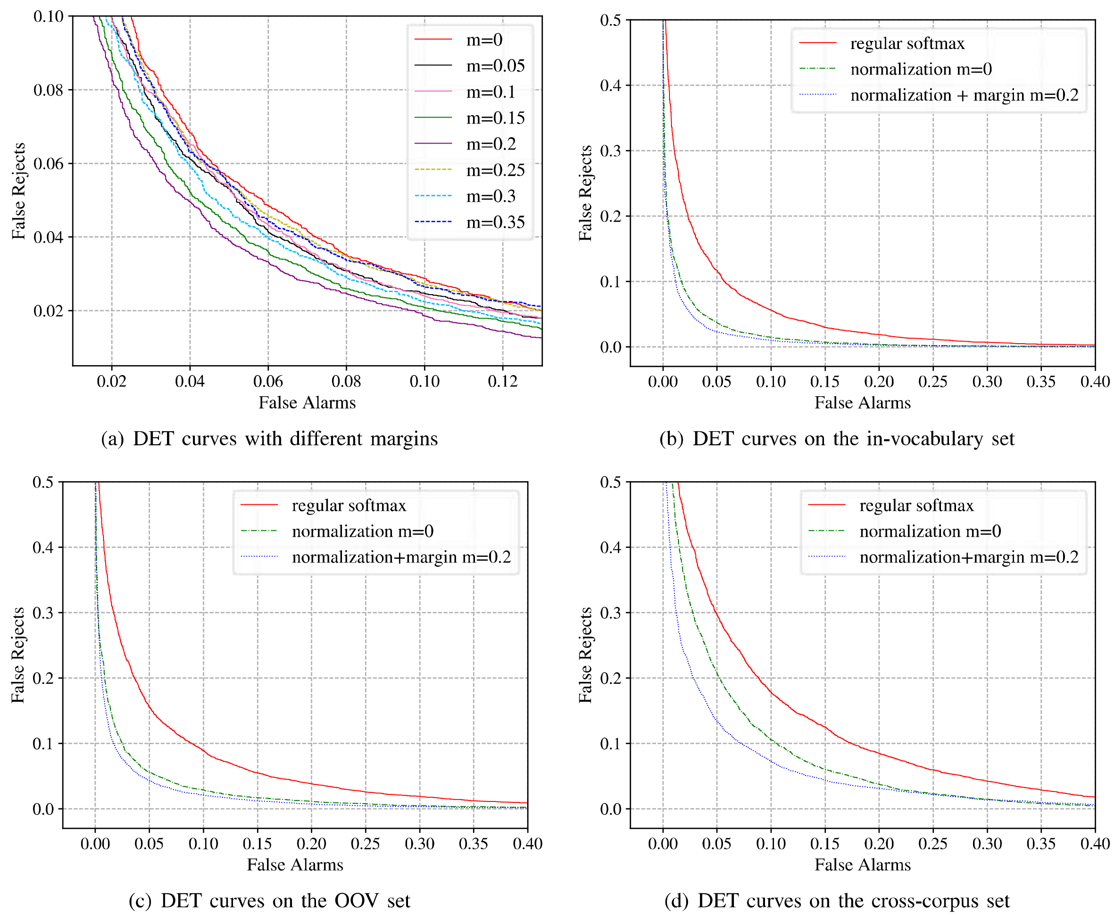
In-vocabulary test set: It has 20,000 segment pairs whose labels have occurred in the training set.

OOV test set: It has 20,000 segment pairs whose labels are out of that 5000 labels.

Cross-corpus test set: It has 20,000 segment pairs and comes from HelloNPU corpus that used for wake-up word detection, a related case of keyword spotting. We choose the keyword "Hello xiaogua" and other speech utterances that do not contain the keyword. Each positive pair involves the keyword "Hello xiaogua" from two different speech utterances and each negative pair involves the keyword "Hello xiaogua" and another non-keyword utterance.

B. Experimental setup

We employ the ResNet in our experiments, which consists of 6 blocks. Each block contains two 3×3 convolution layers with batch normalization. The architecture is illustrated on the right of contrast. We extract the 40-dimensional Mel-frequency cepstrum coefficients (MFCCs) of input audio and pad them to 99 frames. Thus, the input size is 99×40. After normalization, the ResNet generates the 45-dimensional BNF. We use Adam optimizer to train the ResNet for 25 epochs and the initial learning rate is set to 0.1. After each epoch, if the improvement of accuracy on development set is less than 1%, the learning rate is reduced by a factor of 0.7. The mini-batch size is 32. Since the network is more difficult to converge to the



(a) DET curves with different margins (b) DET curves on the in-vocabulary set

(c) DET curves on the OOV set (d) DET curves on the cross-corpus set

Fig. 4. DET curves of the experiment results.

optimal values with hyper-parameter m, we increase the hyper-parameter m linearly after each training epoch. Until the 15th epoch, we fix it to the target margin.

C. The comparison of different margins

We firstly investigate how margin m affects the performance of QbE keyword spotting. We select several representative values of margin $m = [0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35]$ to train the network and evaluate them on the OOV test set. $m = 0$ means that we just employ the normalization for both feature and weight vectors without cosine margin. $m \neq 0$ means we employ both normalization and additive margin. The result is plotted in Figure 4(a). From the detection error tradeoff (DET) curves, we can see that from $m = 0.05$ to $m = 0.2$, the larger the margin, the better the performance. And the best performance is achieved when $m = 0.2$. Although $m = 0.1$ performs similarly to $m = 0.05$ and $m = 0.25, 0.3, 0.35$ performs worse than $m = 0.2$. All the nonzero values contribute to an improvement in QbE keyword spotting tests more or less compared to the zero value. This is because that the margin m reduces the original $\cos\theta$, making the network learn more compact feature representations of the same classes.

Then, we choose the best performance margin value $m = 0.2$ and compare it to the regular softmax loss function on all of the 3 test sets. Furthermore, we also study the benefits of the normalization and margin. So $m = 0, 0.2$ and regular softmax are all displayed in the figures. For convenience, we call $m = 0$ as "normalization" and call $m = 0.2$ as "normalization + margin".

D. The effect of normalization

From Figure 4(b), Figure 4(c), and Figure 4(d), we can the "normalization" performs better than the network trained with regular softmax on all of the test sets. For example, at false alarm rate (FAR) of 2%, the FRR of "normalization" relatively reduces by 60.20% on in-vocabulary set, 57.97% on OOV set, and 20.93% on cross-corpus set.

E. The effect of normalization with additive margin

Figure 4(b), Figure 4(c), and Figure 4(d) also demonstrate the improvement of "normalization + margin". The performance achieves increasingly better from regular softmax to "normalization", and from "normalization" to "normalization + margin" on all of the test sets. In terms of "normalization + margin", at FAR of 2%, the FRR of "normalization + margin" relatively reduces by 79.86% on in-vocabulary set, 68.03% on OOV set, and 46.60% on cross-corpus set compared to the regular softmax. All the relative reductions of "normalization + margin" are more than "normalization", especially on the cross-corpus set. Thus, additive margin can further improve the performance of networks on the basis of normalization.

F. Discussions

From the above experiments, our proposed methods show improvement over the conventional regular softmax for keyword spotting. These are attributed to two reasons. First, normalization helps neural networks focus on angular optimization that is more compatible to the test metric. Comparing to the regular softmax, which implicitly learns features from both Euclidean norm and angle, normalization eliminates the variations in Euclidean norm and constrains the features on hypersphere. Second, the margin m helps to reduce the intra-class distance and, moreover, leads to more discriminative feature learning. Figure 4(b), Figure 4(c), and Figure 4(d) all show that "normalization" outperforms the regular softmax, and "normalization + margin" further improves the performance of "normalization".

V. Conclusions

This paper introduces the normalization operation and additive margin into QbE keyword spotting tasks to learn discriminative embedding features. These have an intuitive geometric interpretation and simple to execute in practice. From the experiments on AISHELL-1 dataset and HelloNPU corpus, normalization with additive margin achieves significant improvements compared to the regular softmax loss. Besides, the performance on cross-corpus test set demonstrates its robustness. At the FAR of 2%, it reduce 46.60% relative FRR in comparison with the regular softmax. In the future, we will find a more stable method to train AM-softmax.

Acknowledgment

This work is supported by the National Key Research Development Plan of China (No.2018YFB1005003), the National Natural Science Foundation of China (NSFC) (No.61425017, No.61831022, No.61771472, No.61603390), the Strategic Priority Research Program of Chinese Academy of Sciences (No.XDC02050100), and Inria-CAS Joint Research Project (No.173211KYSB20170061).

References

[1] X. Anguera and M. Ferrarons, "Memory efficient subsequence dtw for query-by-example spoken term detection," in *2013 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2013, pp. 1–6.

[2] G. Mantena, S. Achanta, and K. Prahallad, "Query-by-example spoken term detection using frequency domain linear prediction and non-segmental dynamic time warping," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 5, pp. 946–955, 2014.

[3] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams," in *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2009, pp. 398–403.

[4] K. Levin, A. Jansen, and B. Van Durme, "Segmental acoustic indexing for zero resource keyword search," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5828–5832.

[5] Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, and L. Lee, "Unsupervised learning of audio segment representations using sequence-to-sequence recurrent neural networks," in *Proc. Interspeech*, 2016, pp. 765–769.

[6] S. Settle and K. Livescu, "Discriminative acoustic word embeddings: Tecurrent neural network-based approaches," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 503–510.

[7] H. Lim, Y. Kim, Y. Kim, and H. Kim, "Cnn-based bottleneck feature for noise robust query-by-example spoken term detection," in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2017, pp. 1278–1281.

[8] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4950–4954.

[9] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5236–5240.

[10] C.-W. Ao and H.-y. Lee, "Query-by-example spoken term detection using attention-based multi-hop networks," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6264–6268.

[11] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220.

[12] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[13] Y. Yuan, C.-C. Leung, L. Xie, H. Chen, B. Ma, and H. Li, "Learning acoustic word embeddings with temporal context for query-by-example speech search," *arXiv preprint arXiv:1806.03621*, 2018.

[14] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[15] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2593–2601.

[16] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai, "Triplet-center loss for multi-view 3d object retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1945–1954.

[17] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling matters in deep embedding learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2840–2848.

[18] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: l 2 hypersphere embedding for face verification," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 1041–1049.

[19] Y. Zheng, D. K. Pal, and M. Savvides, "Ring loss: Convex feature normalization for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5089–5097.

[20] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5265–5274.

[21] 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*, 2016, pp. 770–778.

[22] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[23] Z. Chen, Z. Xie, W. Zhang, and X. Xu, "Resnet and model fusion for automatic spoofing detection." in *INTERSPEECH*, 2017, pp. 102–106.

[24] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[25] S. Wang, J. Hou, L. Xie, and Y. Hao, "Hellonpu: A corpus for small-footprint wake-up word detection research," in *National Conference on Man-Machine Speech Communication*. Springer, 2017, pp. 70–79.

[26] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[27] A. Jansen, S. Thomas, and H. Hermansky, "Weak top-down constraints for unsupervised acoustic model training," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8091–8095.