Two-Dimensional Embeddings for Low-Resource Keyword Spotting Based on Dynamic Time Warping

Kevin Wilkinghoff [1], *Alessia Cornaggia-Urrigshardt* [1], *Fahrettin Gökgöz*

Fraunhofer Institute for Communication, Information Processing and Ergonomics FKIE, Wachtberg, Germany
Email: {kevin.wilkinghoff,alessia.cornaggia-urrigshardt,fahrettin.goekgoez}@fkie.fraunhofer.de

Abstract

State-of-the-art keyword spotting systems consist of neural networks trained as classifiers or trained to extract discriminative representations, so-called embeddings. However, a sufficient amount of labeled data is needed to train such a system. Dynamic time warping is another keyword spotting approach that uses only a single sample of each keyword as patterns to be searched and thus does not require any training. In this work, we propose to combine the strengths of both keyword spotting approaches in two ways: First, an angular margin loss for training a neural network to extract two-dimensional embeddings is presented. It is shown that these embeddings can be used as features for dynamic time warping, outperforming cepstral features even when very few training samples are available. Second, dynamic time warping is applied to cepstral features to turn weak into strong labels and thus provide more labeled training data for the two-dimensional embeddings.

1 Introduction

The goal of keyword spotting (KWS) is to detect specific keywords from a small application-dependent set of words in audio recordings. Therefore, large parts of the audio recordings are not of interest because they do not contain these keywords. In contrast to large-vocabulary continuous speech recognition systems that transcribe all words being spoken and thus are also able to detect a specified subset of keywords, keyword spotting systems require less labeled training data and less computational power. This is advantageous for speech recognition applications with a restricted vocabulary where only limited resources in terms of data and computational power are available. Examples are rarely spoken languages for which only insufficient labeled data is available [1], and difficult (acoustic) environments where labeled data recorded under the same acoustic conditions and also computational power is restricted [2].

An unsupervised approach to detect keywords in audio recordings is to use dynamic time warping (DTW) [3, 4]. Its fundamental idea is to search for a set of provided keywords represented as features that do not require any training in an audio file. This is called query-by-example KWS and has the advantage that no training is needed and it thus works with very limited amounts of data. When at least some labeled data is available, many KWS systems are trained to extract discriminative representations, so-called embeddings, from audio recordings. In [5], it has been shown that supervised embeddings extracted with linear discriminant analysis and graph embeddings outperform unsupervised embeddings and DTW-based KWS systems. Other works utilize different types of neural networks to extract embeddings, for example (Siamese) convolutional neural networks (CNNs) [6], CNNs trained by minimizing an additive angular margin loss [7] or long-short term memory networks (LSTMs) [8, 9]. Using a simple sliding window in combination with these discriminative embeddings has been shown to outperform unsupervised DTW-based approaches. However, most of these systems, especially LSTM-based ones, require much labeled training data in order to work well. In [9], 1115 hours of speech and in [8] 2500 hours of speech are used for training the LSTM-based embeddings. To circumvent this problem, Menon et al. used only 34 minutes of labeled speech to generate labels with a DTW-based approach for a larger unlabeled dataset, which then was used to train a CNN [1]. Another drawback of using supervised embeddings is that a fixed-sized sliding window is less accurate than DTW when prediciting the exact on- and offset of detections. This is especially problematic for applications where sequences of keywords and their order need to be recognized. Examples are telephone numbers, PINs or more generally passwords or -phrases, coordinates or spelled words.

The goal of this work is to combine the strengths of both KWS approaches. To this end, an angular margin loss for learning two-dimensional embeddings and a DTW-based system utilizing these embeddings for spotting sequences of keywords are proposed. The presented system is evaluated on an internal dataset recorded at Fraunhofer FKIE consisting of spoken coordinates in German and a very small training set of less than 7 minutes. It is shown that the presented two-dimensional embeddings outperform cepstral features, more precisely human factor cepstral coefficients (HFCC-ENS) [4]. In additional experiments, cepstrum-based DTW is used to automatically convert weak into strong labels by predicting on- and offset of keywords that are known to be present in a recording. When providing these automatically labeled data as additional training data, the performance obtained with the two-dimensional embeddings is significantly improved.

2 Methodology

2.1 Two-dimensional AdaCos loss

Embeddings obtained by minimizing an angular margin loss function such as ArcFace [10] have been shown to outperform other embeddings and yield state-of-the-art classification performances. The angular margin loss AdaCos [11] does not depend on any hyperparameters to be tuned but uses an adaptive scale parameter while performing equally well as ArcFace. Thus, we propose to modify AdaCos to learn two-dimensional embeddings suitable for being used with DTW.

Let $x_i \in \mathbb{R}^{T_i \times D}$ be an embedding belonging to keyword k of the $K \in \mathbb{N}$ keywords with time dimension T_i and feature dimension D. In standard one-dimensional AdaCos, mean values are learned for each class such that the cosine similarities of all samples to their corresponding class

[1] These authors contributed equally.

means are maximized while the cosine similarities to other class means are minimized. Moreover, a margin between the classes is ensured. For two-dimensional AdaCos, a set of $T \in \mathbb{N}$ embeddings $\mathcal{E}_k \subset \mathbb{R}^D$ instead of a single mean embedding is learned for each keyword. Then, the cosine angle $\theta_{i,k} \in [0, \pi]$ between x_i and $\mathcal{E}_k$ is defined through

$$\cos\theta_{i,k} = \frac{1}{T_i} \sum_{t=1}^{T_i} \max_{e_j \in \mathcal{E}_k} \frac{\langle x_{i,t}, e_j \rangle}{\|x_{i,t}\|_2 \|e_j\|_2}. \quad (1)$$

This definition of the angle is the only difference of the two-dimensional AdaCos to the regular one-dimensional loss. Therefore, the probability of sample x_i belonging to keyword j is still given by

$$P_{i,j} := \frac{\exp(\tilde{s} \cdot \cos\theta_{i,j})}{\sum_{k=1}^{K} \exp(\tilde{s} \cdot \cos\theta_{i,k})} \quad (2)$$

with the standard adaptive scale parameter $\tilde{s}$ (see [11]).

To ensure that all audio samples of keywords used to train the neural network are of length 0.5 seconds, shorter samples are zero-padded and a sliding window of size 0.5 seconds and a step size of 0.1 seconds is used for longer samples. To obtain two-dimensional embeddings for the test sentences, which are much longer than 0.5 seconds, first a sliding window of size 0.5 seconds with a hop size of $1/T$ is used to compute another embedding for each window position. Then, all embeddings belonging to a single test recording are combined into one longer embedding by taking the mean of all $1 \times D$-dimensional vectors of the embeddings for which their corresponding windows overlap at a given position.

To obtain a database consisting of a single representation for each keyword, the DTW barycenter averaging (DBA) algorithm [12] as implemented in [13] is used. Instead of computing a regular Euclidean mean of all samples belonging to a single keyword, which minimizes the Euclidean distance, this algorithm estimates the more general Fréchet mean that minimizes the distance implied by DTW. As a result, the examples stored in the keyword database are more similar to corresponding keyword samples when searching for a keyword with DTW and thus the performance is improved.

2.2 Network architecture

The architecture of the neural network used for extracting two-dimensional embeddings is shown in Tab. 1. It consists of a modified ResNet architecture [14] in combination with the 2D AdaCos loss and, except for the loss function, strongly resembles the architecture used in [15]. Each residual block includes *batch normalization* layers [16] and *LeakyReLu* [17] with $\alpha = 0.1$ as nonlinearities. As input to the neural network, *log-Mel spectrograms* with 64 Mel-bins, a window size of 1024 and a hop size of 256 are extracted from raw waveforms of length 0.5 seconds, resulting in features of size 32×64. To avoid overfitting of the model to the limited amount of training data, two data augmentation techniques are used: 1) *SpecAugment* [18], which consists of frequency masking, time masking and time warping, and 2) *mixup* [19], more precisely *manifold mixup* [20], for random linear interpolations between hidden representations of training data. To take the usage of mixup into account when minimizing the two-dimensional AdaCos loss while training, the extension presented in [15] is used. In all experiments, the network, implemented in Tensorflow [21], is trained for 1000 epochs with a batch size of 32 using *Adam* [22] for optimization.

layer name	structure	output size
input	-	32×64
residual block	$\begin{pmatrix} 3\times3 \\ 3\times3 \end{pmatrix} \times 2$, stride= 1×1	$32 \times 64 \times 16$
residual block	$\begin{pmatrix} 3\times3 \\ 3\times3 \end{pmatrix} \times 2$, stride= 1×2	$32 \times 32 \times 32$
residual block	$\begin{pmatrix} 3\times3 \\ 3\times3 \end{pmatrix} \times 2$, stride= 1×2	$32 \times 16 \times 64$
residual block	$\begin{pmatrix} 3\times3 \\ 3\times3 \end{pmatrix} \times 2$, stride= 1×2	$32 \times 8 \times 128$
max pooling	1×8, stride= 1×1	32×128
dense (representation)	linear	32×32
2D AdaCos	-	38

Table 1: Modified ResNet architecture used for extracting two-dimensional embeddings.

2.3 Voice activity detection (VAD)

A pre-processing step for analyzing long audio recordings or a post-processing step to refine the results of the proposed KWS system (cf. Sec. 2.4) is a *Voice Activity Detection* (VAD) algorithm. We propose a VAD based on a combination of *spectral energy* and two particular audio features, namely *spectral flux* and *spectral flatness*. The use of energy is restricted to pre-defined frequency subbands, one for low frequencies in the range of 100-1000 Hz focusing on the tonal components of speech and neglecting low-frequency noise, and one for higher frequencies in the range of 5500-8000 Hz to consider e.g. sibilants. In addition, two audio features, which have been successfully used for speech detection tasks ([23–25]), are used to capture the particular characteristics of speech signals. These features are combined by thresholding the corresponding feature curves. Thresholds were fixed empirically by evaluating the training data. The performance of the VAD evaluated on manually labeled training data has a TP-rate of 99.46% and a FP-rate of 0.77%.

2.4 DTW-based keyword spotting (KWS)

In the proposed query-by-example approach, every *keyword* and target sentence is represented by a sequence of feature vectors, a 2D-feature matrix, obtained from the previously derived *embeddings*, allowing a frame-wise comparison of the target signal with the trained keyword patterns. Let $\mathtt{K}_i \in \mathcal{K}$ be the feature representation of keyword i from the keyword set $\mathcal{K}$ and $\mathtt{S} \in \mathcal{S}$ be the corresponding feature representation of a target signal. To account for the different lengths of keyword utterances $\mathtt{K}_i$ and the sequences $\mathtt{S}$, we apply sub-sequence DTW [26, 27] to align keywords with sub-sequences of the target signal. In classical DTW approaches, two sequences of feature vectors are time-aligned by calculating a pair-wise similarity between all the feature vectors, thus obtaining a cost matrix $\mathtt{C}$, which is transformed into an accumulated cost matrix $\mathtt{D}$ following pre-defined step size conditions. Sub-sequence DTW applies the same technique but returns several possible matchings by extracting local maxima of the resulting accumulated cost matrix. As opposed to *diagonal matching*, the step size condition of classical DTW is extended to larger steps, in our case to $\{(2,1),(1,1),(1,2)\}$, to account for faster and slower speakers. $\mathtt{C}$ is calculated using the inner product of any two feature vectors j and k,

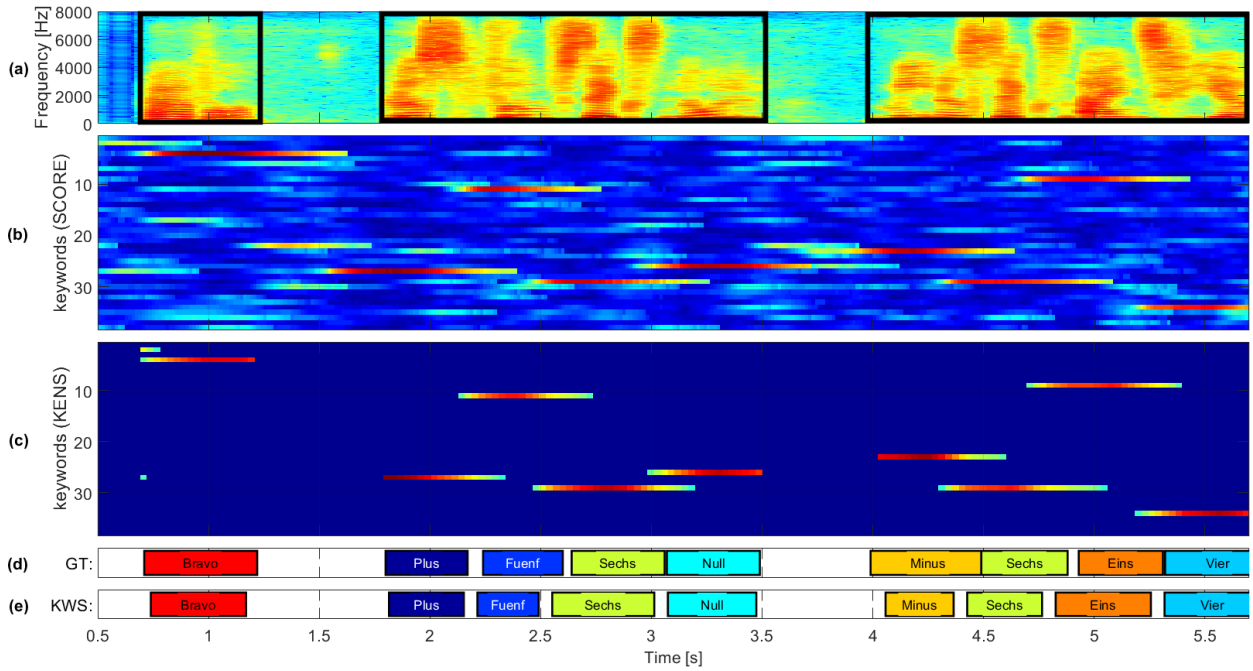

Figure 1: DTW-KENS-KWS: *Dynamic Time Warping Keyword Energy Normalized Statistics Keyword Spotting* showing (a) the spectrogram of a target signal (with VAD results indicated by black boxes), (b) the score matrix $\mathtt{S}_{DTW}$, (c) the enhanced KENS matrix, (d) a manually generated ground truth annotation, and (e) the keyword spotting results.

i.e. $Sim^{\mathtt{K}_i,\mathtt{S}}(j,k) := \langle v_j^{\mathtt{K}_i}, v_k^{\mathtt{S}} \rangle / \|v_j^{\mathtt{K}_i}\|_2 \|v_k^{\mathtt{S}}\|_2$, where $v_j^{\mathtt{K}_i}$ denotes the j^{th} feature vector of a keyword $\mathtt{K}_i$ and $v_k^{\mathtt{S}}$ the k^{th} the one of a target sentence (cf. [4]). Then $\mathtt{C} = 1 - Sim^{\mathtt{K}_i,\mathtt{S}}$.

2.5 DTW-KENS-KWS

Instead of extracting paths on the basis of an accumulated cost matrix for each keyword $\mathtt{K}_i$, we only consider the last row of $\mathtt{D}$, which can be interpreted as a cost curve and, by reverting the negation, as a *score* function $\mathtt{s}^i_{DTW}$ for a keyword $\mathtt{K}_i$. All $\mathtt{s}^i_{DTW}$ are concatenated to a *score matrix* $\mathtt{S}_{DTW}$ for all $\mathtt{K}_i \in \mathcal{K}$ and a given signal $\mathtt{S} \in \mathcal{S}$, where high values indicate likely matchings. An example of $\mathtt{S}_{DTW}$ is given in Fig. 1 (b), where red colors indicate a high score. The y-axis shows the index of the keywords (cf. 3.1).

To enhance the high score regions, we apply a technique from music processing used to calculate CENS features [28, 29], which has also been adopted for keyword spotting [4] (cf. Sec. 3.2), consisting of *normalization* and *quantization*, *smoothing*, and *downsampling*. These operations are applied to the DTW-based score matrix representing *keywords*. Hence we refer to the resulting sequence of post-processed feature vectors by *Keyword Energy Normalized Statistics* (KENS). An example of this KENS matrix is shown in Fig. 1 (c).

Keyword spotting is performed by picking the column-wise maxima of the KENS matrix and extracting subblocks satisfying predefined lengths. Larger blocks are divided into consecutive matches of the same keyword by considering personalized average lengths. In addition, the proposed VAD (Sec. 2.3) is used to adjust the start and end times of the detected keywords. The VAD results are indicated by the black boxes in Fig. 1 (a). Furthermore, the results can be post-processed by using knowledge about the data such as in the given case of spoken coordinates where letters only appear at the beginning of blocks of numbers.

3 Experimental Results

3.1 Dataset

The dataset used for all evaluations contained in this paper is an internal dataset recorded by Fraunhofer FKIE. The corpus consists of read coordinates in the German language. All recordings are downsampled to 16kHz and the annotations have been manually verified twice. The training subset of the dataset consists of 37 keywords and an additional class *Silence* consisting of all six background noise files of Google Speech Commands [30]. The keywords are: *Acht*, *Alpha*, *Bravo*, *Charlie*, *Delta*, *Drei*, *Echo*, *Eins*, *Foxtrot*, *Fünf*, *Fünnef*, *Golf*, *Hotel*, *Ich berichtige*, *India*, *Juliett*, *Kilo*, *Korrektur*, *Korrigiere*, *Lima*, *Mike*, *Minus*, *Neun*, *November*, *Null*, *Plus*, *Quebec*, *Sechs*, *Sieben*, *Sierra*, *Tango*, *Victor*, *Vier*, *Whiskey*, *X-ray*, *Yankee*, *Zwo*. These keywords are all read at most once by 12 male and 7 female speakers, 3 of which are non-native. The recordings were done in different office rooms using standard sound pressure microphones. As a result, there are 16 to 19 versions of each keyword for training with a total duration of less than 7 minutes. For validation and testing, the same speakers read up to 201 sentences each, resulting in a validation set with a duration of 179 minutes containing 1723 sentences and a test set with a duration of 240 minutes containing 2090 sentences.

3.2 Comparison of features

As a first experiment, it is shown that the proposed two-dimensional embeddings outperform classical cepstral features. HFCC-ENS features have been shown to outperform Mel-frequency cepstral coefficients (MFCCs) when detecting keywords with DTW [4]. Therefore, we only consider the former. HFCC-ENS are computed by applying particular filterbanks optimized for human audio perception to spectral feature vectors and then post-processed by nor-

feature	loss	WER validation set	WER test set
HFCC-ENS	–	0.2533	0.2660
embedding	flatten + softmax	0.3118	0.3115
embedding	2D AdaCos	0.2213	0.2100

Table 2: Word error rates obtained with different features.

malization, quantization, smoothing, and downsampling – a technique applied also in our proposed KWS algorithm (Sec. 2.5). In our evaluation, we use these features together with sub-sequence DTW to extract keywords. The results are post-processed by exploiting domain knowledge, in particular using keyword lengths and allowing letters only at the beginning of number clusters. As opposed to the KENS *embeddings* approach, several examples of the same personalized keyword are used for the DTW KWS method.

The resulting WERs can be found in Tab. 2. Recall that the training dataset is very small and only consists of less than 7 minutes of speech. Still, the two-dimensional embeddings extracted with the proposed 2D AdaCos result in significantly lower WERs than HFCC-ENS features. To verify that the loss is actually important, the network architecture was altered by replacing the 2D AdaCos loss with a flattening operation in combination with a softmax loss. This led to a much higher WER, even higher than the one obtained with HFCC-ENS features.

3.3 Turning weak into strong labels

When using data-driven models such as neural networks, their true power comes from using as much high-quality training data as possible. Creating weak labels for an unlabeled dataset is much less time-consuming than strong labeling because experts can simply listen once and write down all words while listening instead of stopping for each word and marking the precise start- and endpoints. However, weakly labeled data cannot be used to train the presented neural network because it is only known what is being said in a sentence but not when the actual words begin or end. Hence, we propose to use DTW with cepstral features to automatically convert weak into strong labels and thus create additional training data. Since prior knowledge in the form of weak labels is available, this procedure should lead to training data of higher quality than when creating strong labels from scratch as done in [1].

To this end, all the keywords $\mathtt{K}_i$, given by a fixed feature representation $\mathcal{F}(\mathtt{K}_i)$ – in our case HFCC-ENS, are concatenated in the order they appear in a given training sentence $\mathtt{S}$, also transformed into the corresponding feature representation $\mathcal{F}(\mathtt{S})$, resulting in a super-feature matrix $\mathcal{F}(\mathtt{K})$. This matrix is then aligned with the sequence of feature vectors of $\mathtt{S}$, providing a global start and end time of the best alignment of the super-keyword $\mathtt{K}$. Knowing the length of each *sub*-keyword $\mathtt{K}_i$ allows to sub-divide the matching result. In order to evaluate the performance of the automatic segmentation, which is shown in Fig. 2, we consider the overlap between each resulting segment and its ground truth. This overlap can be interpreted as a percentage of the result interval (x-axis) and of the ground truth (y-axis). The colors indicate the percentage of segments which are considered correct (in %/100). One example is the white point on Fig. 2: 90% of the segments can be seen as correct, if we require the overlap of a ground truth segment with the corresponding result segment to be

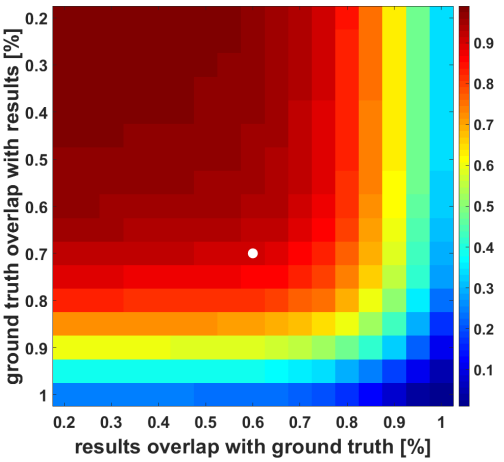

Figure 2: Performance of the automatic segmentation.

automatically labeled data used for: neural network	Fréchet mean	WER
		0.2100
	✗	0.1841
✗		0.0724
✗	✗	0.0720

Table 3: Word error rates obtained on the test set using automatically labeled data as additional training data.

at least 70% of the length of the ground truth and 60% of the length of the result segment – which for the purpose of data augmentation may be sufficient. As a result, the total length of the training dataset is increased from less than 7 minutes to 95 minutes.

The effects of using automatically labeled data as additional training data can be found in Tab. 3. As expected, the WER decreases significantly when using more data for training. This is especially true for the neural network, where the WER is reduced by a factor of 3. Hence, the two-dimensional embeddings improve in representing the keywords as features when more training data is available.

4 Conclusions and Future Work

In this work, a keyword spotting system capable of detecting sequences of keywords in low-resource settings has been presented. The system is based on two-dimensional embeddings obtained by training a neural network with a novel 2D AdaCos loss and utilizes these embeddings as features for DTW-based keyword spotting. In experiments conducted on a coordinate recognition dataset in German with a training set of less than 7 minutes, it has been shown that the features lead to a lower WER than HFCC-ENS features. Furthermore, a procedure for converting weak into strong labels has been proposed and shown to significantly improve the performance of the two-dimensional embeddings by generating more training data.

For future work, it can be investigated whether using soft-DTW [31] inside the AdaCos layer leads to an improved performance of the two-dimensional embeddings when using DTW. Furthermore, it is planned to conduct experiments where the two-dimensional embeddings are used in combination with prototypical networks [32] for few-shot learning.

References

[1] R. Menon, H. Kamper, J. Quinn, and T. Niesler, "Fast ASR-free and almost zero-resource keyword spotting using DTW and CNNs for humanitarian monitoring," in *19th Annual Conference of the International Speech Communication Association (Interspeech)*, pp. 2608–2612, ISCA, 2018.

[2] H.-C. Schmitz, F. Kurth, K. Wilkinghoff, U. Müllerschkowski, C. Karrasch, and V. Schmid, "Towards robust speech interfaces for the ISS," in *International Conference on Intelligent User Interfaces (IUI) Companion*, pp. 110–111, ACM, 2020.

[3] M. D. Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. V. Compernolle, "Template-based continuous speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 15, no. 4, pp. 1377–1390, 2007.

[4] D. Von Zeddelmann, F. Kurth, and M. Müller, "Perceptual audio features for unsupervised key-phrase detection," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 257–260, IEEE, 2010.

[5] K. Levin, K. Henry, A. Jansen, and K. Livescu, "Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings," in *Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 410–415, IEEE, 2013.

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

[7] H. Ma, Y. Bai, J. Yi, and J. Tao, "Hypersphere embedding and additive margin for query-by-example keyword spotting," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 868–872, IEEE, 2019.

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

[9] J. Hou, L. Xie, and Z. Fu, "Investigating neural network based query-by-example keyword spotting approach for personalized wake-up word detection in mandarin chinese," in *10th International Symposium on Chinese Spoken Language Processing, (ISCSLP)*, pp. 1–5, IEEE, 2016.

[10] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4690–4699, IEEE, 2019.

[11] X. Zhang, R. Zhao, Y. Qiao, X. Wang, and H. Li, "AdaCos: Adaptively scaling cosine logits for effectively learning deep face representations," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10823–10832, IEEE, 2019.

[12] F. Petitjean, A. Ketterlin, and P. Gançarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern recognition*, vol. 44, no. 3, pp. 678–693, 2011.

[13] R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar, and E. Woods, "Tslearn, a machine learning toolkit for time series data," *Journal of Machine Learning Research*, vol. 21, no. 118, pp. 1–6, 2020.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, IEEE, 2016.

[15] K. Wilkinghoff, "Sub-cluster AdaCos: Learning representations for anomalous sound detection," in *International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2021.

[16] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *32nd International Conference on Machine Learning (ICML)*, vol. 37, pp. 448–456, 2015.

[17] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *30th International Conference on Machine Learning (ICML)*, 2013.

[18] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *20th Annual Conference of the International Speech Communication Association (Interspeech)*, pp. 2613–2617, ISCA, 2019.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations (ICLR)*, 2018.

[20] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, and Y. Bengio, "Manifold mixup: Better representations by interpolating hidden states," in *36th International Conference on Machine Learning (ICML)*, vol. 97 of *Proceedings of Machine Learning Research*, pp. 6438–6447, PMLR, 2019.

[21] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 265–283, 2016.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[23] Y. Ma and A. Nishihara, "Efficient voice activity detection algorithm using long-term spectral flatness measure," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2013, no. 1, pp. 1–18, 2013.

[24] S. Urrigshardt, S. Kreuzer, and F. Kurth, "General detection of speech signals in the time-frequency plane," in *ITG Symposium on Speech Communication*, pp. 1–5, VDE, 2016.

[25] S. Lee, J. Kim, and I. Lee, "Speech/audio signal classification using spectral flux pattern recognition," in *Workshop on Signal Processing Systems*, pp. 232–236, IEEE, 2012.

[26] F. Kurth and D. von Zeddelmann, "An analysis of mfcc-like parametric audio features for keyphrase spotting applications," *ITG Symposium on Speech Communication*, 2010.

[27] D. von Zeddelmann, F. Kurth, and M. Müller, "Vergleich von Matching-Techniken für die Detektion gesprochener Phrasen," *Deutsche Jahrestagung für Akustik*, pp. 257–260, 2010.

[28] M. Müller, F. Kurth, and M. Clausen, "Chroma-based statistical audio features for audio matching," in *Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 275–278, IEEE, 2005.

[29] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *6th International Conference on Music Information Retrieval (ISMIR)*, pp. 288–295, 2005.

[30] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *CoRR*, vol. abs/1804.03209, 2018.

[31] M. Cuturi and M. Blondel, "Soft-DTW: a differentiable loss function for time-series," in *34th International Conference on Machine Learning, (ICML)*, pp. 894–903, PMLR, 2017.

[32] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 4077–4087, 2017.