GENERATING TTS BASED ADVERSARIAL SAMPLES FOR TRAINING WAKE-UP WORD DETECTION SYSTEMS AGAINST CONFUSING WORDS

Haoxu Wang[1,2]*, *Yan Jia*[2]*, *Zeqing Zhao*[3], *Xuyang Wang*[3], *Junjie Wang*[3], *Ming Li*[1,2]†

[1]School of Computer Science, Wuhan University, Wuhan, China
[2]Data Science Research Center, Duke Kunshan University, Kunshan, China
[3]AI Lab of Lenovo Research, Beijing, China

ming.li369@duke.edu

Abstract

Wake-up word detection models are widely used in real life, but suffer from severe performance degradation when encountering adversarial samples. In this paper we discuss the concept of confusing words in adversarial samples. Confusing words are commonly encountered, which are various kinds of words that sound similar to the predefined keywords. To enhance the robustness of the wake-up word detection system against confusing words, we propose several methods to generate the adversarial confusing samples for simulating real confusing words scenarios in which we usually do not have any real confusing samples in the training set. The generated samples include concatenated audio, synthesized data, and partially masked keywords. Moreover, we use a domain embedding concatenated system to improve the performance. Experimental results show that the adversarial samples generated in our approach help improve the system's robustness in both the common scenario and the confusing words scenario. In addition, we release the confusing words testing database called HI-MIA-CW for future research.

1. Introduction

In intelligent speech processing applications, the Keyword Spotting (KWS) system, including wake-up word detection, plays an important role in human-computer interaction. KWS aims to detect a predefined keyword or a set of keywords in a continuous audio stream. Studies have been proposed to deliver robust approaches with high detection accuracy. The authors of [1] adopted dynamic time warping (DTW) for keyword spotting back in 1985, then hidden Markov models (HMM) [2, 3, 4], deep neural networks (DNN)[5, 6, 7] and other various neural network structures including convolutional neural networks (CNN) [8], temporal convolutional neural networks [9, 10], recurrent (RNN) neural networks[11, 12] and Transformer[13] have also been proposed for this task. However, the probability of false alarm becomes higher under complex acoustic environments and ambiguous content. Without further adaptation, KWS system may misclassify fillers as keywords since some of the filler actually sound close to the keywords. Those are the adversarial samples, which are called confusing words (CW). Moreover, it is expensive to acquire human recorded adversarial samples for training a KWS system that can accurately classify confusing words, especially when the keywords are customized by the users.

In this paper, we discuss the concept of confusing words. And we will also release a supplemental database called HI-MIA-CW which was compiled following the same setup of the HI-MIA[14] database to record.There are about 16k audios with 12 confusion words patterns in table 1 for the keyword in HI-MIA database from new 30 speakers. This data is included in our evaluation set. Then we propose several methods to generate some adversarial samples for simulating real confusing words employed on an end-to-end approach to address the aforementioned issue. The idea is motivated by the maximum mutual information (MMI) criterion to improve the discriminative power of the model[15]. The first technique is to concatenate the waveform of the real subword audio. The second adversarial samples augmentation is performed by a text-to-speech (TTS) system. The use of synthesized speech for data augmentation is also not new[16, 17, 18]. [19, 20] shows that synthetic data can help train the keyword spotting model. The third method is applying random masking on speech signals to simulate confusing words like keyword audio that are interrupted by mute in the middle. Moreover, we use domain embeddings extracted from pre-trained LSTM domain classifier to help overcome the domain shift problems. To the best of our knowledge, this is the first research where the concept of confusing words in KWS scenarios is discussed and we explore the augmentation methods to generate adversarial samples of confusing words to improve the performance of the wake-up word detection system. Both augmentation methods achieve significant improvement on the end-to-end KWS model. Especially for TTS augmentation, the false rejects rate drops from 68.60% to 9.81% at twenty false alarms in one hours compared with the one that is evaluated with the system trained without the aforementioned confusing words augmentation approaches.

This rest of the paper is organized as follows. Section 2 discusses the confusing words and the released database. Section 3 describes the framework of the CNN based KWS system. Section 4 presents our augmentation methods. Section 5 discusses the experimental results, and the conclusion is provided in section 6.

2. Confusing words

The definition of confusion words changes with the application scenario. In the natural language processing (NLP) scenarios, some confusing words with similar meanings, but different spellings and pronunciations, will appear in similar contexts and some other confusing words are misspelled. [21] presents

*The first two authors contributed equally to this work.
†Corresponding Author.

a system named Automatic Confusion words Extraction (ACE), which takes a Chinese word as input and automatically outputs its easily misspelled confused words.

Unlike confusion words in the NLP domain, confusion words in the speech domain are those words that sound similar to predefined keywords or are the part of the keywords. Let's take the keyword "ni hao, mi ya"(Hello Mia) as an example. Based on the idea of similar pronunciation and fragmentation, we came up with the following confusing words in table 1. Table 1 shows the phoneme sequence of the keyword and confusion words, where the subscript of phoneme represents tone. Confusion words as adversarial samples attack the acoustic model in the wake-up word system, cause false rejections that severely disrupt the usage experience.

In order to compare the performance of models in practical applications, we used the same setup of HI-MIA database [14] to further record 16k audios with the same 12 confusion word patterns in table 1 from new 30 speakers. The supplemental database HI-MIA-CW is released.[1]

Table 1: Phoneme sequences of the keyword and confusion words

Data Type	Words	Phoneme Sequence
Keyword	ni hao, mi ya	N I_2 H A_3 U_3 M I_3 YH I_4 A_4
Confusion Words	ni hao mi	N I_2 H A_3 U_3 M I_3
	ni hao, ni hao	N I_2 H A_3 U_3 N I_2 H A_3 U_3
	ni hao ya	N I_2 H A_3 YH I_4 A_4
	hao mi ya	H A_3 U_3 M I_3 YH I_4 A_4
	ni mi ya	N I_2 M I_3 YH I_4 A_4
	ni hao	N I_2 H A_3 U_3
	mi ya, mi ya	M I_3 YH I_4 A_4 M I_3 YH I_4 A_4
	hao mi, hao mi	H A_3 U_3 M I_3 H A_3 U_3 M I_3
	ni hao mi	N I_2 H A_3 U_3 M I_1
	hao mi ya	H A_3 U_3 M I_1 YH I_4 A_4
	mi ya, mi ya	M I_1 YH I_4 A_4 M I_1 YH I_4 A_4
	hao mi, hao mi	H A_3 U_3 M I_1 H A_3 U_3 M I_1

3. Model architecture

In this section, we present our baseline system, which is modified from the CNN-based KWS system [8]. As shown in Figure 1, our baseline system consists of three modules:(i) a feature extraction module, (ii) a convolutional neural network and (iii) a posterior processing module.

The feature extraction module converts the audio signals into acoustic features. 80 dimensional log-mel filterbank features are extracted from a speech frame with 50ms long and 12.5ms shift. Then we apply a segmental window with 121 frames to generate training samples that contain enough context information as the input of the model.

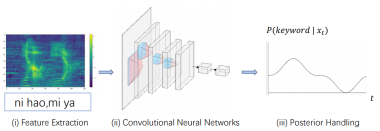

Figure 1: Framework of the baseline system.

Our backbone network consists of three convolutional layers each followed by a maximum pooling layer. For all three CNN layers, the kernel size is set to (3,3), the stride is (1,1), and the pooling size is set to (2,2). Two fully connected layers and a final softmax activation layer are applied as the back-end prediction module to obtain the keyword occurrence probability.

The acoustic feature sequence is transformed into a posterior probability sequence of selected keywords by the model. We perform the keyword detection algorithm over a sliding window with length T_s. Here we use $\mathbf{x}^{(i)} = \{x_i, x_{i+1}, \ldots, x_{i+T_s}\}$ to denote one input window over the segment X that contains N frames. Then the keyword confidence score is calculated as follows:

$$conf(X) = \max_{1 \le t \le N - T_s} P_{keyword}(\mathbf{x}^{(t)}) \quad (1)$$

where $P_{keyword}(x^{(t)})$ is the posterior probability of the keyword appearing in the window started at frame t. The KWS system triggers once when the confidence score exceeds a predefined threshold.

4. Adversarial samples

Models that perform well on a test data set might fail in real life applications where many testing samples are confusion words. This problem becomes more important in the case of customized wake up words defined by the users. In this case, to reduce the performance degradation when applying KWS in unmatched scenarios and improve the robustness of KWS, we propose three methods to generate adversarial samples for confusion words.

4.1. Waveform Concatenation

To obtain training samples of confused words, it is natural to use unit selection and waveform concatenation.[22] shows the difference between concatenative and neural TTS system. We use a Large Vocabulary Conversational Speech Recognition(LVCSR) to align the audio and the text in a labeled public speech dataset, then truncate the audio to get waveform of each subword of the keyword. Truncated audio may come from different speakers. We simply concatenate the waveform according to the order of the subwords in keywords and confused words to generate the adversarial samples.

4.2. Text-to-speech Augmentation

We obtain synthesized data from a mandarin multi-speaker TTS system [23]. In this setup, 7k speakers from publicly available datasets and internal datasets are collected and used for synthesis. For each speaker, we first extract the speaker embedding with one utterance by using the TTS system. Then 3 kinds of synthesized samples are generated by the multi-speaker TTS system conditioned on the speaker embedding: (i) positive samples that whose content is the keywords (ii) negative samples that do not contain keywords, (iii) adversarial negative samples that are confusion words that have contents close to the keywords.

4.3. Masked Audio

We applied random masking on keyword samples and used them as the adversarial negative data in training to improve the robustness of our KWS model. The KWS model should yield undetected results when having these masked samples since masked samples simulate confusion words like keyword audio that are interrupted by mute in the middle. For each positive sample, we generate corresponding masked samples online by

[1] http://www.openslr.org/120/

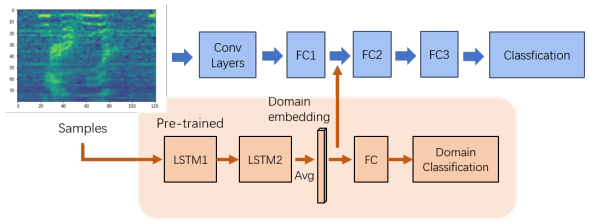

Figure 2: Framework of the domain embedding system.

replacing 40%-60% audio signals with Gaussian white noise, unlike SpecAug[24] which uses the mean value.

4.4. Domain Adaptation

Based on the assumption that the distribution of synthetic data and real data are different, we incorporate the domain adaptation method of Environmental Domain Embeddings with TTS augmentation in the training step in order to improve the robustness of the model and make it fit the distribution of real data better. As shown in Figure 2, we train the KWS system with the domain embedding derived from a pre-trained domain classifier. The method is inspired by [25] and [26] which applied the domain embedding to incorporate domain knowledge into network training and improved the performance of the keyword classifier on far-field conditions.

The domain classifier is trained by samples from different domain which include real domain, synthetic domain and concatenated domain. The domain classifier consists of two stacked LSTM layers followed by an average pooling layer and a final fully-connected linear layer. Domain embeddings are extracted from the output of the pooling layer and its dimension is fixed to 128. The domain classifier is trained before the training of CNNs. When we train the CNN model, we extract the domain embedding from the pre-trained domain classifier and concatenate the embedding to the output of the first fully-connected layer. Then the concatenated features are fed into two linear layers for predicting the posterior probability of the keyword.

Table 2: Dataset statistics (P: positive, N: negative)

Samples	Label	Train	Test
HI-MIA	P	23k	2k
AISHELL-1	N	105k	10k
HI-MIA-CW	N	-	16k
Concatenated Keywords	P	23k	-
Concatenated Confusion Words	N	23k	-
Synthesized Keywords	P	7k	-
Synthesized Confusion Words	N	90k	-
Synthetic Negative	N	188k	-
Masked	N	23k	-

5. Experimental results

5.1. Dataset

Natural speech recorded by native speakers and generated adversarial samples are both used for training in our experiments. For natural speech data, the HI-MIA dataset [14] is used as the positive samples. The HI-MIA dataset includes speech data recorded by one close-talking microphone and six 16-channel circular microphone arrays. Each utterance contains content with four Chinese characters “ni hao, mi ya” (Hello, Mia). We only use the recordings from the single-channel close-talking microphone. Samples from 300 randomly selected speakers are used as the training set, and samples from 30 speakers are used as the HI-MIA test set. The AISHELL-1 [27] dataset is used as the negative sample of real speech data. Utterances from 300 speakers are selected for training, and utterances from 30 speakers are used as the AISHELL-1 test set.

For concatenated data, each subword in keyword contains about 3k samples, and we concatenate the waveform online to synthesize keywords (concat-wake) and confusion words (concat-cw) as train set. For synthetic data, we have 7k different utterance samples from all speakers that are synthesized according to the keyword text. They are used as synthetic positive keywords (synt-wake) train set. And we also have samples that include 12 confusion word patterns with 90k different voices, where utterance from all speakers are used as the synthetic confusion words (synt-cw) train set. Also we mask the postive samples online as negative samples according to the Section 4 (mask). In addition, 188k negative audio samples are synthesized with provided text from AISHELL-2 [28] (synt-neg). In order to compare the performance of models in practical applications, we also used the HI-MIA-CW database as the test set(real-cw). The statistics of the data we used for training and testing is shown in table 2, where the term ‘Real’ denotes natural speech, including utterances from HI-MIA database (Real Positive), AISHELL-1 (Real Negative) and HI-MIA-CW (Real Confusion Words).

5.2. Experimental Setup

We preprocess the HI-MIA training set by trimming the beginning silence and force align the audio by a speech recognition system trained on the AISHELL-2 dataset. For each sample, we obtain the start time of pronouncing the word ”ni” and use the following 121 frames as the final input, where 121 frames are enough for speaking the keyword according to the alignment information. Our models are trained for 100 epochs with Nesterov momentum Stochastic gradient descent optimizer. The initial learning rate of the optimizer is set to 0.1 and decays when the training loss has not decreased for several rounds. During evaluation, we have a sliding window with a frame length of 121 for each utterance and detect the occurrence of the expected keyword.

Six KWS systems are trained and evaluated regarding different training setups in our experiments: (i) **baseline**: use all real samples (include Real Positive set and the Real Negative set), which are shown in table 2, for training. (ii) **real+concat-***: use the all real samples, all concatenated samples(include concat-wake and concat-cw) for training. (iii) **real+syn-***: use all real samples, all synthetic samples(include synt-wake, synt-cw and synt-neg) for training. (iv) **real+mask**: use all real samples, and the masked samples for training. (v) **real+concat-*+syn-*+mask**: use all real samples, all concatenated samples, all synthetic samples, and the masked samples for training. (vi) **real+concat-*+syn-*+mask+EMB**: use all real samples, all concatenated samples, all synthetic samples, and the masked samples for training. Pre-trained Domain Classifier is incorporated in this setup.

There are two combination sets for evaluation: (i) **real**: use the test set from Real Positive set and Real Negative set for evaluation. (ii) **real + real-cw**: in addition to the test sets mentioned above, the natural samples of confusion words (real-CW) are also included.

5.3. Results

Results are shown in Figures 3, 4 and Tables 3, 4, where Figure 3 and Table 3 show the performance of models on real test sets (HI-MIA + AISHELL-1) without confusing words testing samples, Figure 4 and Table 4 show performance of models on the real + real-cw test set. As for real test sets, we choose the false rejection rate under one false alarm per hour as each model's performance criterion separately. Table 4 presents the KWS performance of the five models regarding the false rejection rate when the false alarm rate per hour is 20, as adding confusion words in the test set makes the task more challenging.

Table 3: Performances of models trained with different methods on the real test sets (the false rejection (FR) rate (%) under one false alarm (FA) per hour)

Training set	real
baseline	0.417
real + concat-*	1.67
real + syn-*	1.37
real + mask	**0.334**
real + concat-* + synt-* + mask	3.29
real + concat-* + synt-* + mask + EMB	0.523

Table 4: Performances of models trained with different methods on the real+real-cw test sets (the false rejection (FR) rate (%) under twenty false alarms per hour)

Training set	real+real-cw
baseline	68.60
real + concat-*	46.05
real + syn-*	38.54
real + mask	63.63
real + concat-* + synt-* + mask	16.87
real + concat-* + synt-* + mask + EMB	**9.81**

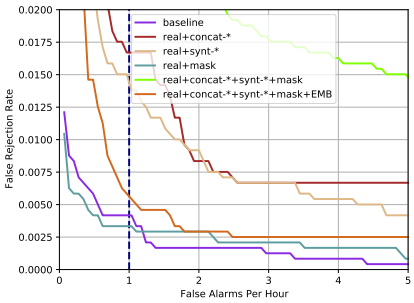

Figure 3: Performances of models on the real test sets

From Table 3 and 4 we can obtain the following observations. First, the baseline system performed well in real test sets without confusing word samples. However, the performance of the baseline system degrades dramatically on confusion words examples, which happens frequently in real-life applications. Second, directly adding adversarial samples leads to performance degradation on the real test set but masked samples help train the system and achieve the best result (0.334) on the real test set. Moreover, after the domain embedding algorithm is applied, the system also maintains the performance on

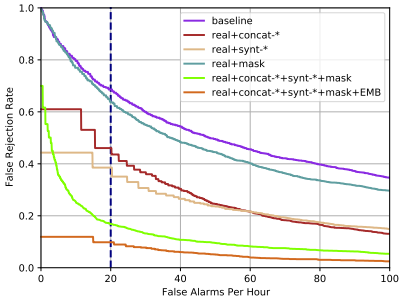

Figure 4: Performances of models on the real+real-cw test sets

the real test set and achieves the result (0.523). It is because domain adaptation methods help the system to learn to fit the distribution of real data better and overcome the degradation in performance due to the domain shift. Third, the accuracy on the confusion word test set has been significantly improved by adding adversarial synthetic samples. It can also be found that adding concatenated samples and adding masked samples does not improve the performance as much as TTS synthesized ones. This may be due to insufficient simulation of confusion word when masked samples are added separately, while possibly misleading the system to learn whether there is a Gaussian distribution of judgments. Also the concatenated samples show steep changes in the splicing breakpoints in the spectral features and do not simulate the confusion words well enough. Fourth, adding synthetic samples along with concatenated and masked samples can help the system to better learn the difference between confusing words and keywords, which achieve the result 16.87% on the real+real-cw testing set.

Finally, comparing to the baseline without any augmentation, this augmentation setup with the domain adaptation method achieves best performance on the real+real-cw testing set and shows great robustness on confusion words scenarios as the false rejection rate under twenty false alarms per hour decreases from 68.60% to 9.81%.

6. Conclusions

In this paper, we discuss the concept of the confusion words and focus on the task of small-footprint keyword spotting in this scenario, then show the effectiveness of generating adversarial samples to train a keyword recognition system. Confusing words that sound very similar to the keywords lead to a significant degradation in system performance. We release the supplemental database HI-MIA-CW and adopt three augmentation strategies to enhance the robustness, including concatenated samples, masked samples and synthesized samples with the domain adaptation methods. Experimental results show that our proposed methods can effectively maintain the accuracy on general real test data and at the same time achieve significant improvement under the test condition with confusing word samples.

7. Acknowledgments

This research is funded in part by the National Natural Science Foundation of China (62171207), Science and Technology Program of Guangzhou City (202007030011) and Lenovo. Many thanks for the computational resource provided by the Advanced Computing East China Sub-Center.

8. References

[1] Alan Higgins and R Wohlford, "Keyword recognition using template concatenation," in *Proc. ICASSP*, 1990, pp. 1233–1236.

[2] R. C. Rose and D. B. Paul, "A hidden markov model based keyword recognition system," in *Proc. ICASSP*, 1990, pp. 129–132.

[3] J. G. Wilpon, L. R. Rabiner, C. . Lee, and E. R. Goldman, "Automatic recognition of keywords in unconstrained speech using hidden markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 1870–1878, 1990.

[4] P. Modi, L. Miller, and J. Wilpon, "Improvements and applications for key word recognition using hidden markov modeling techniques," in *Proc. ICASSP*, 1991, pp. 309–312.

[5] Ming Sun, D. Snyder, Yixin Gao, Varun K. Nagaraja, Mike Rodehorst, S. Panchapagesan, N. Strom, Spyridon Matsoukas, and Shiv Vitaladevuni, "Compressed time delay neural network for small-footprint keyword spotting," in *Proc. Interspeech*, 2017, pp. 3607–3611.

[6] S. Panchapagesan, Ming Sun, Aparna Khare, Spyridon Matsoukas, A. Mandal, Björn Hoffmeister, and Shiv Vitaladevuni, "Multi-task learning and weighted cross-entropy for dnn-based keyword spotting," in *Proc. Interspeech*, 2017, pp. 760–764.

[7] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *Proc. ICASSP*, 2014, pp. 4087–4091.

[8] Tara N Sainath and Carolina Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech*, 2015.

[9] Seungwoo Choi, Seokjun Seo, Beomjun Shin, Hyeongmin Byun, Martin Kersner, Beomsu Kim, Dongyoung Kim, and Sungjoo Ha, "Temporal Convolution for Real-Time Keyword Spotting on Mobile Devices," in *Proc. Interspeech 2019*, 2019, pp. 3372–3376.

[10] Somshubra Majumdar and Boris Ginsburg, "MatchboxNet: 1D Time-Channel Separable Convolutional Neural Network Architecture for Speech Commands Recognition," in *Proc. Interspeech 2020*, 2020, pp. 3356–3360.

[11] Santiago Fernández, Alex Graves, and Jürgen Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Proc. ICANN*, 2007, pp. 220–229.

[12] Martin Woellmer, Bjoern Schuller, and Gerhard Rigoll, "Keyword spotting exploiting long short-term memory," *Speech Communication*, pp. 252–265, 2013.

[13] Yiming Wang, Hang Lv, Daniel Povey, Lei Xie, and Sanjeev Khudanpur, "Wake word detection with streaming transformers," in *Proc. ICASSP*. 2021, pp. 5864–5868, IEEE.

[14] Xiaoyi Qin, Hui Bu, and Ming Li, "Hi-mia: A far-field text-dependent speaker verification database and the baselines," in *Proc. ICASSP*, 2020, pp. 7609–7613.

[15] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi.," in *Proc. Interspeech*, 2016, pp. 2751–2755.

[16] Luise Valentin Rygaard, "Using synthesized speech to improve speech recognition for low–resource languages," vol. 8, pp. 2018, 2015.

[17] Masato Mimura, Sei Ueno, Hirofumi Inaguma, Shinsuke Sakai, and Tatsuya Kawahara, "Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition," in *Proc. SLT*, 2018, pp. 477–484.

[18] Ramazan Gokay and Hulya Yalcin, "Improving low resource turkish speech recognition with data augmentation and tts," in *2019 16th International Multi-Conference on Systems, Signals Devices (SSD)*, 2019, pp. 357–360.

[19] James Lin, Kevin Kilgour, Dominik Roblek, and Matthew Sharifi, "Training keyword spotters with limited and synthesized speech data," in *Proc. ICASSP*, 2020, pp. 7474–7478.

[20] Yiling Huang, Yutian Chen, Jason Pelecanos, and Quan Wang, "Synth2aug: Cross-domain speaker recognition with tts synthesized speech," in *Proc. SLT*, 2021, pp. 316–322.

[21] Chun-Hung Wang, Jason S Chang, and Jian-Cheng Wu, "Automatic chinese confusion words extraction using conditional random fields and the web," in *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, 2013, pp. 64–68.

[22] Michelle Cohn and Georgia Zellou, "Perception of concatenative vs. neural text-to-speech (TTS): differences in intelligibility in noise and language attitudes," in *Interspeech 2020*. 2020, pp. 1733–1737, ISCA.

[23] Zexin Cai, Chuxiong Zhang, and Ming Li, "From Speaker Verification to Multispeaker Speech Synthesis, Deep Transfer with Feedback Constraint," in *Proc. Interspeech*, 2020.

[24] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[25] Suyoun Kim, Bhiksha Raj, and Ian R. Lane, "Environmental noise embeddings for robust speech recognition," *CoRR*, vol. abs/1601.02553, 2016.

[26] Haiwei Wu, Yan Jia, Yuanfei Nie, and Ming Li, "Domain aware training for far-field small-footprint keyword spotting," *Proc. Interspeech*, pp. 2562–2566, 2020.

[27] Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *Proc. O-COCOSDA*, 2017, pp. 1–5.

[28] Jiayu Du, Xingyu Na, Xuechen Liu, and Hui Bu, "Aishell-2: transforming mandarin asr research into industrial scale," *arXiv preprint arXiv:1808.10583*, 2018.