Interspeech 2022
18-22 September 2022, Incheon, Korea

Keyword Spotting with Synthetic Data using Heterogeneous Knowledge Distillation

Yuna Lee, Seung Jun Baek

Korea University, Republic of Korea
{ylee5813,sjbaek}@korea.ac.kr

Abstract

It is crucial that Keyword Spotting (KWS) systems learn to understand new classes of user-defined keywords, which however is a challenging task requiring high-quality audio datasets. We propose KWS with Heterogeneous Embedding Knowledge Distillation (HEKD) which uses only synthetic data of unseen keyword classes. In HEKD, a reference model transfers the heterogeneous knowledge on seen classes to the student model for classifying keywords of unseen classes. By mimicking the embedding function of reference model trained on real data via a contrastive learning approach, we show that student model can learn to discriminate unseen keyword classes guided by synthetic data. In addition, we propose to maximize the dispersion of embedding clusters of unseen keywords with approximation guarantees in order to enhance the inter-class variability. Experiments show that HEKD outperforms baseline schemes using few-shot learning and those pre-trained on a large volume of data, demonstrating its effectiveness and efficiency.

Index Terms: keyword spotting, contrastive learning, audio synthesis, speech embeddings, zero-shot learning

1. Introduction

Keyword Spotting (KWS) has been actively explored with applications to on-device voice assistant services such as Google Assistant, Apple Siri, etc. [1]. As the demands on personalized services arise, the task of learning user-defined keywords, such as buzzwords and jargons which were unseen by the device, became important for KWS systems. Most of KWS studies [2, 3, 4, 5, 6] have focused on lightweight models which regard the entire unseen keywords as belonging to unknown class. Since gathering training data for user-defined keywords requires much time and effort, there have been attempts to leverage models pretrained on a large-scale audio dataset [7] for customized keyword classification through fine-tuning or transfer learning. In the case of transfer learning, several works [8, 9, 10] have utilized contrastive learning on large-scale audio datasets. Although these methods have made great progresses, they did not directly tackle the data-scarcity problem of unseen classes [11]. A number of studies attempted to maximize the utilization of a limited amount of data, e.g., Metric Learning [12], Meta Learning [13, 14], and self-supervised learning [15]. Some studies have used synthesized data obtained by Text-to-Speech (TTS) methods [16, 17, 18], although they focused mostly on binary classification which distinguishes keywords from non-keywords. Moreover, domain gap between the synthetic and real domains [19] resulted in limited performances in classifying samples in real audio.

In this paper, we propose Heterogeneous Embedding Knowledge Distillation (HEKD) to train KWS networks for unseen classes with synthetic data only. The idea is depicted in Fig. 1. There is a reference model f pretrained on seen class-

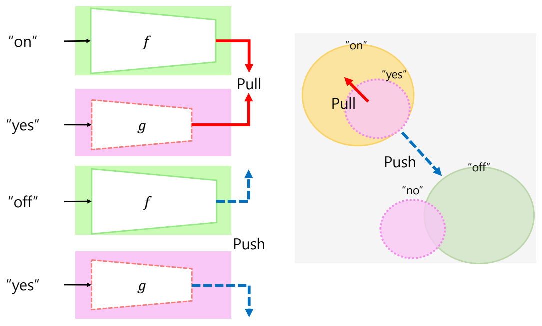

Figure 1: *{"on","off"} are keywords in seen classes. {"yes","no"} are keywords in unseen classes. "yes" is paired with "on", and student model g learns to assimilate the embeddings of "yes" to those of "on" guided by reference model f.*

es of real audio, whereas our goal is to train student model g in synthesized audio. In HEKD, reference f transfers heterogeneous knowledge for discriminating seen classes to student g for discriminating unseen classes. As in Fig. 1, unseen class "yes" and seen class "on" are *paired.* g learns to embed "yes" in the way f embed "on". Similarly if g embeds "no" (unseen) in the way f embeds "off" (seen, paired with "no"). g can discriminate "yes" and "no", because f is already trained to discriminate "on" and "off". Such knowledge on class separation was bestowed to f through *real* audio. We show that g can also perform reasonably well in real audio by mimicking the embedding function of f, even though the training of g is guided by synthetic data. We take a contrastive approach, i.e., paired classes "pull" and unpaired classes "push" as in Fig. 1, in transferring heterogeneous knowledge. In this way, g can emulate the embedding function of f with different classes of data.

Our scheme uses synthetic data only, thus can be regarded as a zero-shot method, because virtually any keyword can be synthesized through TTS. In this respect, our work is comparable to Data-Free Learning (DAFL) [20] in that it uses data synthesis to train smaller models under the supervision of a trained teacher network. While DAFL focuses on transferring knowledge of seen classes to the student model by synthesizing data on the fly, we focus on training the model by transferring heterogeneous knowledge from seen classes to unseen classes.

Moreover, we propose Maximum Dispersion Pairing, which is a 2-approximation algorithm for making the clusters of unseen keywords, whose locations are learned from seen classes, as dispersed as possible in the embedding space so as to improve the inter-class variability. We also propose Contrastive Heterogeneous Distillation, by which the forming of clusters of unseen keywords are guided by those of seen keywords in a

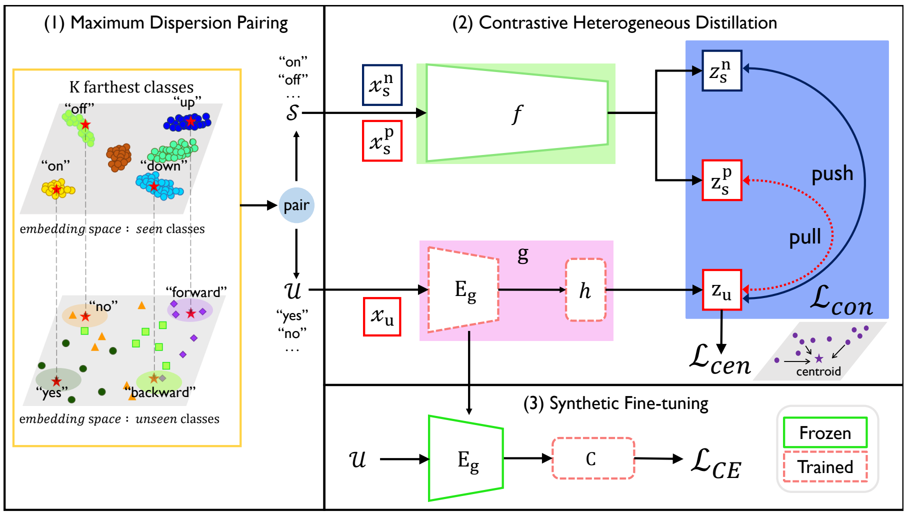
(1) Maximum Dispersion Pairing
K farthest classes
"off"
"up"
"on"
"down"
embedding space : seen classes
"forward"
"no"
"yes"
"backward"
embedding space : unseen classes
pair
(2) Contrastive Heterogeneous Distillation
f
g
h
push
pull
centroid
(3) Synthetic Fine-tuning
C
Frozen
Trained

Figure 2: *Overview of HEKD framework. $\mathcal{S}$ and $\mathcal{U}$ denote synthetic dataset of seen and unseen classes respectively. (1) Maximum Dispersion Pairing: K farthest centroids of seen classes are selected, and unseen and selected seen classes are matched as pairs. (2) Contrastive Heterogeneous Distillation: given embedding output z_u from student model g of an unseen keyword, positive embedding z_s^p and z_u are pulled together while negative embedding z_s^n is pushed away. (3) Synthetic Fine-tuning: after the heterogeneous distillation, the encoder of g is used as fixed backbone to train a classifier C on synthetic dataset $\mathcal{U}$ of unseen classes.*

contrastive manner. Through experiments we compare HEKD with schemes trained on a small number of real audio samples, i.e., a few-shot setting, or embedding models pretrained on large amount of data. We show that HEKD outperforms these baselines, demonstrating the effectiveness in bridging the domain gap between real and synthesized data.

2. Methods

2.1. Outline

Our framework uses two models: reference model f pretrained on seen classes, and student model g to be trained for unseen classes. The number of seen and unseen classes are denoted by N and K respectively. Reference model f is regarded as a "black box" whose weights are "frozen" during the entire learning process. f is pre-trained only on N seen classes of keywords in real audio. Typically f would represent relatively bigger models handling a larger number (N) of classes . By contrast, student function g represents smaller models for classifying less (K) classes of user-defined keywords, with small memory footprint suitable for portable devices and applications. Thus we assume that $K \leq N$.

Since f is trained only on the actual audio data of seen classes, it can cause overfitting or negative transfer [11] if unlearned classes are trained through existing methods such as fine-tuning [21] or transfer learning [22]. Besides, if g is trained with synthetic data only, the performance will be poor due to the domain gap between the synthetic and real domains [19]. We intend to train g such that g can emulate the feature embedding function f learned from real data in the embedding space although f is trained for different keyword classes.

Our strategy is as follows: for each unseen class c_u, we find seen class c_s, such that (c_s, c_u) becomes a matched pair. *We let g learn to embed class-c_u keywords in the way f embeds class-c_s keywords.* Thus, the transfer of knowledge from f to g is for different classes of data where f and g can be different types of models as well, which we call *heterogeneous embedding knowledge distillation*. Suppose g would like to discriminate keywords in, say class $c'_u \neq c_u$ from those in c_u. Suppose (c'_s, c'_u) is a matched class pair. Since keyword clusters from c_s and c'_s are well-separated by f, those of c_u and c'_u will be also well-separated in the embedding space of g, if g can mimic the class separation from f. Moreover, f was trained to separate c_s and c'_s based on the real data in those classes. This will make c_s and c'_s to be further separated compared to the case where f were trained only on synthetic datasets, because in that case clusters c_s and c'_s would have occupied smaller region in the embedding space [19], or the clusters of synthetic data can even be *included* in the clusters of real data embeddings, because real data provides more general representation of audio. g can leverage such large separation to discriminate c_u and c'_u, leading to improved classification performance.

Thus, our goal is to create matched (seen,unseen) pairs of classes, and train g guided by f. It is guaranteed that each unseen class has always a matching pair because $K \leq N$. The following issue need to be addressed in the design:

(a) The (seen,unseen) class pairs should be matched such that, K clusters of matched classes of seen keywords are as far apart as possible, in order to maximize inter-class variability in unseen classes.

(b) Student g should be trained such that the embeddings of keywords of classes in a matched pair should be close, and those from unmatched pairs should be far apart.

To that end, we propose HEKD which consists of two parts: (a) Maximum Dispersion Pairing (MDP), (b) Contrastive Heterogeneous Distillation.

2.2. Maximum Dispersion Pairing

Our goal is to choose K out of N seen classes of keywords, such that those K keyword clusters of embeddings are located as far as possible from each other. Firstly we find a set of N cluster centroids, one for each seen class. To do so, we generate a number of synthesized keywords in seen classes, and compute the centroids of outputs from f averaged over the generated data. Let T denote the set of computed centroids, where T contains N embedding vectors. We would like to find subset $S \subset T$ which solves the following optimization problem:

$$\begin{aligned} \text{maximize} \quad & \min_{u,v \in S} d(u,v) \\ \text{subject to} \quad & S \subset T,\ |S| = K \end{aligned}$$

where $d(u, v)$ denotes the Euclidean distance between points u and v. This combinatorial problem is equivalent to *discrete p-dispersion problem* [23] which is known to be NP-hard. Thus we will use a greedy algorithm proposed by [24] as follows. We begin by $S = \{u, v\}$ where u and v are two centroids in T which has the maximum inter-distance. Next we update S incrementally by iteratively adding a centroid which maximizes the minimum distance to the centroids in S until $|S| = K$. The greedy algorithm was shown to be a factor 2 approximation to discrete p-dispersion [24]. Once we find S, we randomly match unseen classes to the seen classes associated with the centroids in S. Pseudocode of MDP is provided in Algorithm 1.

Algorithm 1 Maximum Dispersion Pairing

```
 1: Input: Synthetic Dataset $\mathcal{D}_c$ for seen class $c = 1, \cdots, N$,
    pretrained model $f$
 2: Output: Set $S$ of $K$ centroids with maximum dispersion
 3: $S \leftarrow \emptyset, T \leftarrow \emptyset$,
 4: for each class $c = 1, \cdots, N$ do
 5:     $T \leftarrow T \cup \{\mathbb{E}[f(\mathcal{D}_c)]\}$          ▷ Calculate centroids
 6: $u^*, v^* = \mathrm{argmax}_{u,v \in T}\, d(u,v)$
 7: $S \leftarrow \{u^*, v^*\}$                                        ▷ Initialize $S$
 8: while $|S| < K$ do                                               ▷ Greedy update of $S$
 9:     $u^* = \mathrm{argmax}_{u \in T \setminus S}\, [\min_{v \in S} d(u,v)]$
10:     $S \leftarrow S \cup \{u^*\}$
```

2.3. Contrastive Heterogeneous Distillation

We perform Contrastive Heterogeneous Distillation, in which reference model f will transfer the knowledge of separating keywords of seen classes to student model g with respect to unseen classes. f and g will take different inputs from seen and unseen classes respectively during the training of g. We will take a contrastive learning approach as follows.

As shown in Figure 2, g consists of encoder $E_g(\cdot)$ and projection head $h(\cdot)$. The purpose of h is to match the dimensions of the output embeddings from f and g. $h(\cdot)$ is implemented as a 2-layer Multi-Layer Perceptron (MLP). Consider a synthetic sample (x_u, y_u) from unseen classes where x_u denotes a synthesized keyword and y_u denotes its class. Let the output embedding vector from g by z_u, i.e., $z_u = g(x_u)$. Next, consider synthetic sample (x_s, y_s) from seen classes. The output embedding from reference model, or $f(x_s)$, is considered a positive embedding with respect to z_u if classes y_s and y_u is a pair matched through MDP, and a negative embedding otherwise.

We adopt contrastive loss function $\mathcal{L}_{con}$ to minimize (resp. maximize) the differences in embeddings from matched (resp. unmatched) class pairs. Let the output embeddings from f of positive sample and negative sample in seen classes be z_s^p and z_s^n. We use Supervised Contrastive (SupCon) loss [25] for every unseen classes embedding vector z_u to *pull* positive samples z_p^s while *push* negative samples z_n^s away in the embedding space:

$$\mathcal{L}_{con} = -\sum_{u \in B} \sum_{z_s^p} \log \frac{\exp(z_u \cdot z_s^p / \tau)}{\sum_{z_s^n} \exp(z_u \cdot z_s^n / \tau)} \qquad (1)$$

where $\cdot$ denotes the dot product, B is the minibatch , and hyperparameter $\tau = 0.1$ denotes the temperature.

Our framework have connections with prior methods on contrastive domain generalization [26, 19, 27]. Our method differs in that it jointly learns to lessen the disparity between real and synthetic domains and to emulate the embedding function of pretrained models. In addition, our framework allows the student model to be of smaller size than the pretrained reference model.

In order to densely cluster data within the same class, we use center loss [28] to minimize the intra-class variability. The idea is to reduce the distance between each embedding of the class and the centroid of the embeddings of the corresponding class in minibatch. The loss is denoted by $\mathcal{L}_{cen}$ with the embedding vector z_u with center c_{y_u} for class label y_u in minibatch B:

$$\mathcal{L}_{cen} = \frac{1}{2} \sum_{u \in B} \|z_u - c_{y_u}\|_2^2 \qquad (2)$$

The total loss function is given by $\mathcal{L} = \mathcal{L}_{con} + \lambda \mathcal{L}_{cen}$ where hyperparameter $\lambda = 0.1$ balances two losses.

After the heterogeneous distillation, we use the trained E_g as backbone, and train a classifier C using synthesized dataset, which we call *synthetic fine-tuning*. C is a fully-connected layer appended to E_g where E_g is frozen in this stage. The purpose is to separate training processes associated with distillation and classification, i.e., the distillation involving f and g should not be interfered by training a classifier involving g only. We train C with additional synthetic samples of unseen classes, and use cross entropy loss $\mathcal{L}_{CE}$.

3. Experiments

3.1. Experiment setup

Dataset. We conducted our experiments on the synthesized dataset which has the same classes as Google Speech Commands dataset (GSC) v0.01 [29]. We synthesized audio data with pretrained vocoder network [30] by using the Librispeech ASR Corpus dataset [31]. We generated 3,150 audio data per class using 251 speakers' voices. We set 4-way ('right' 'down', 'go', 'one') and 5-way ('cat', 'yes', 'house', 'wow', 'seven') for unseen classes. We consider two cases for seen classes: 10 seen classes as ('marvin', 'up', 'on', 'happy', 'dog', 'eight', 'off', 'five', 'three', 'six'), and ('bed', 'stop', 'left', 'two', 'zero') classes were added for 15 seen classes. We tested our method on unseen classes of test dataset of GSC v0.01.

Baselines. In this paper, we compare our methods with three baselines: FS-KWS [13], MAML-EXT [14], and Embed-Head [32] where FS-KWS and MAML-EXT are based on few-shot learning approaches. FS-KWS exploits the average value of feature embeddings from seen classes, so that samples from unseen class can be inferred to the closest seen class. MAML-EXT applied Model Agnostic Meta Learning [33] approach to adapt unseen classes based on seen classes. The Embed-Head method uses fine-tuning of an embedding model pretrained on 200 million audio clips.

Implementation Details. The input audio is sampled at 16kHZ, and is transformed into a Log-amplitude Mel-Spectrogram tensor after Short-Time Fourier Transform (STFT) is applied. For reference model f, we used ResNet18 and ResNet50 [34] which is pretrained on real data of seen classes of GSC v0.01. For student model g, we used small encoder model ResNet10 and C64

Table 1: *4-way Classification*

Method	Encoder		1-shot	5-shot	10-shot
MAML-EXT [14]	-		55.61%	65.39%	71.74%
FS-KWS [13]	C64		64.96%	84.08%	87.03%
	ResNet10		70.03%	85.80%	89.66%
Embed-Head [32]	-			92.60%	
	f	g	10-seen		15-seen
HEKD (ours)	ResNet18	C64	88.68%		90.04%
		ResNet10	92.01%		**93.71%**
	ResNet50	C64	89.63%		89.30%
		ResNet10	92.49%		92.27%

Table 2: *5-way Classification*

Method	Encoder		1-shot	5-shot	10-shot
MAML-EXT [14]	-		47.48%	61.15%	64.78%
FS-KWS [13]	C64		60.33%	82.90%	83.98%
	ResNet10		68.25%	84.05%	85.70%
Embed-Head [32]	-			89.95%	
	f	g	10-seen		15-seen
HEKD (ours)	ResNet18	C64	88.64%		88.14%
		ResNet10	**92.48%**		92.22%
	ResNet50	C64	90.03%		86.98%
		ResNet10	92.24%		90.51%

[35]. LARS optimizer is used [36] with learning rate of 3.0 and batch size of 2048 for 200 epochs.

3.2. Results

Performance Comparison. We compared performance of HEKD with aforementioned baselines. Table 1 and 2 show the classification accuracy for unseen classes. HEKD shows better accuracy in most cases than the baselines using few-shot learning, i.e., FS-KWS [13] and MAML-EXT [14]. The results show that an appropriate usage of synthetic data can be better than learning through a small amount of real data.

Embed-Head [32] shows good performance in overall for 4-way task. However, HEKD outperforms Embed-Head when there are 15-seen classes with g is based on ResNet10. For 5-way task, however, Embed-Head showed lower accuracy than HEKD in most of the cases. Importantly, Embed-Head was pretrained on 200M real audio clips, thus HEKD can be regarded as a significantly efficient framework for zero-shot keyword spotting. Presumably the head model of Embed-Head trained through fine-tuning with synthetic data, did not properly learn acoustic features of real audio, due to the domain gap between the real and synthetic data [19].

Next we compare the classification performance in 4-way and 5-way cases. Note the performance is expected to be degraded if the number of unseen classes increases from 4 to 5. It was shown that the few-shot learning methods (FS-KWS, META-ext) incurred more severe degradation than zero-shot learning with synthetic data (HEKD and Embed-Head) when the number of unseen classes increases. If we compare HEKD and Embed-Head, the performance degradation of HEKD was lower than that of Embed-Head. This indicates that our method can learn more robustly from synthetic data even if the number of unseen keyword classes increases.

Ablation Study. Firstly we perform ablation study on the Maximum Dispersion Pairing (MDP). From Table 3, we compared MDP algorithm and the random pairing between seen and unseen classes. We observe that MDP algorithm outperforms ran-

Table 3: *Keyword Classification comparison on Pair selection*

	4-way				5-way			
f Encoder	ResNet18				ResNet18			
g Encoder	ResNet10		C64		ResNet10		C64	
Seen classes	10	15	10	15	10	15	10	15
Random	89.15%	88.61%	86.41%	82.11%	88.63%	90.80%	87.27%	84.52%
Ours	**92.01%**	**93.71%**	**88.68%**	**90.04%**	**92.48%**	**92.22%**	**88.64%**	**88.14%**
f Encoder	ResNet50				ResNet50			
Random	90.41%	88.57%	86.83%	81.24%	89.25%	88.51%	87.77%	78.32%
Ours	**92.49%**	**92.27%**	**89.63%**	**89.30%**	**92.24%**	**90.51%**	**90.03%**	**86.98%**

dom pair selection in all cases. We also observe the trend such that, for fixed networks, the difference in performances for 10-seen and 15-seen classes are not high for MDP; however, for random pairing, the performance tends to degrade significantly in case of 15-seen classes. As the number of seen classes increases, the embedding space of the pretrained model becomes more "crowded". That is, the gap between clusters of seen classes becomes narrow, implying that the minimum distance between embedding clusters will decrease. Thus, if we use random pairing, it is more likely that the randomly selected clusters of seen classes are closely located. Consequently, inter-class variability will tend to decrease with random pairing, whereas MDP is robust to such cases, because MDP maximizes the inter-cluster distances with guaranteed approximation to the optimal dispersion of clusters.

Next, we perform ablation study for synthetic fine-tuning. We compared HEKD which is a two-step training process, i.e., firstly contrastively trains g for distillation and secondly uses synthetic fine-tuning for classification, with the scheme which trains g jointly for distillation and classification. On average, the proposed scheme obtained gains in accuracy by 1.29% in 4-way and 2.68% in 5-way classification. Thus, it is better to separate the training processes of distillation and classification in each of which the network is optimized for different purposes.

4. Conclusion

In this paper, we introduced keyword spotting with synthesized samples using heterogeneous embedding knowledge distillation which can learn unseen classes of keywords. The proposed framework contrastively trains the student model to learn the separation of keyword clusters guided by the reference model, however dealing with different classes of data between reference and student models. We also introduced a method of matching unseen classes to seen classes which greedily maximizes the inter-class variability with approximation guarantees. Experiments showed that HEKD outperformed baselines based on few-shot learning, and one based on the embedding backbone pretrained by a large amount of real audio samples, demonstrating its architectural efficiency.

5. Acknowledgement

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Creative Consilience program (IITP-2022-2020-0-01819) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation), and by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. NRF-2021R1A2C1007215) and international cooperation program (NRF-2021K2A9A2A06047061).

6. References

[1] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[2] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5484–5488.

[3] A. Coucke, M. Chlieh, T. Gisselbrecht, D. Leroy, M. Poumeyrol, and T. Lavril, "Efficient keyword spotting using dilated convolutions and gating," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6351–6355.

[4] C. Yang, X. Wen, and L. Song, "Multi-scale convolution for robust keyword spotting." in *INTERSPEECH*, 2020, pp. 2577–2581.

[5] S. Hu, J. Wang, Y. Wang, and W. Yang, "Separable temporal convolution plus temporally pooled attention for lightweight high-performance keyword spotting," *arXiv preprint arXiv:2108.12146*, 2021.

[6] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted Residual Learning for Efficient Keyword Spotting," in *Proc. Interspeech 2021*, 2021, pp. 4538–4542.

[7] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[8] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3875–3879.

[9] A. T. Liu, S.-W. Li, and H.-y. Lee, "Tera: Self-supervised learning of transformer encoder representation for speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2351–2366, 2021.

[10] H. Al-Tahan and Y. Mohsenzadeh, "Clar: contrastive learning of auditory representations," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2530–2538.

[11] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 293–11 302.

[12] J. Huh, M. Lee, H. Heo, S. Mun, and J. S. Chung, "Metric learning for keyword spotting," in *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2021, pp. 133–140.

[13] A. Parnami and M. Lee, "Few-shot keyword spotting with prototypical networks," *arXiv preprint arXiv:2007.14463*, 2020.

[14] Y. Chen, T. Ko, L. Shang, X. Chen, X. Jiang, and Q. Li, "An Investigation of Few-Shot Learning in Spoken Term Classification," in *Proc. Interspeech 2020*, 2020, pp. 2582–2586.

[15] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "Byol for audio: Self-supervised learning for general-purpose audio representation," in *2021 International Joint Conference on Neural Networks, IJCNN 2021*, 2021.

[16] E. Sharma, G. Ye, W. Wei, R. Zhao, Y. Tian, J. Wu, L. He, E. Lin, and Y. Gong, "Adaptation of rnn transducer with text-to-speech technology for keyword spotting," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7484–7488.

[17] X. Zheng, Y. Liu, D. Gunceler, and D. Willett, "Using synthetic audio to improve the recognition of out-of-vocabulary words in end-to-end asr systems," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5674–5678.

[18] A. Werchniak, R. B. Chicote, Y. Mishchenko, J. Droppo, J. Condal, P. Liu, and A. Shah, "Exploring the application of synthetic audio in training keyword spotters," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7993–7996.

[19] W. Chen, Z. Yu, Z. Wang, and A. Anandkumar, "Automated synthetic-to-real generalization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1746–1756.

[20] H. Chen, Y. Wang, C. Xu, Z. Yang, C. Liu, B. Shi, C. Xu, C. Xu, and Q. Tian, "Data-free learning of student networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3514–3522.

[21] T. Bluche and T. Gisselbrecht, "Predicting detection filters for small footprint open-vocabulary keyword spotting," *arXiv preprint arXiv:1912.07575*, 2019.

[22] D. Seo, H.-S. Oh, and Y. Jung, "Wav2kws: Transfer learning from speech representations for keyword spotting," *IEEE Access*, vol. 9, pp. 80 682–80 691, 2021.

[23] E. Erkut, "The discrete p-dispersion problem," *European Journal of Operational Research*, vol. 46, no. 1, pp. 48–60, 1990.

[24] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi, "Heuristic and special case algorithms for dispersion problems," *Operations Research*, vol. 42, no. 2, pp. 299–310, 1994.

[25] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 661–18 673, 2020.

[26] Y. Chen, W. Li, and L. Van Gool, "Road: Reality oriented adaptation for semantic segmentation of urban scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7892–7901.

[27] W. Chen, Z. Yu, S. Mello, S. Liu, J. M. Alvarez, Z. Wang, and A. Anandkumar, "Contrastive syn-to-real generalization," 2021.

[28] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*. Springer, 2016, pp. 499–515.

[29] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[30] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu *et al.*, "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," *Advances in neural information processing systems*, vol. 31, 2018.

[31] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[32] J. Lin, K. Kilgour, D. Roblek, and M. Sharifi, "Training keyword spotters with limited and synthesized speech data," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7474–7478.

[33] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.

[34] 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.

[35] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[36] Y. You, I. Gitman, and B. Ginsburg, "Large batch training of convolutional networks," *arXiv preprint arXiv:1708.03888*, 2017.