DISENTANGLED TRAINING WITH ADVERSARIAL EXAMPLES FOR ROBUST SMALL-FOOTPRINT KEYWORD SPOTTING

Zhenyu Wang[1*] , Li Wan[2], Biqiao Zhang[2], Yiteng Huang[2], Shang-Wen Li[2],
Ming Sun[2], Xin Lei[2], Zhaojun Yang[2]

[1] The University of Texas at Dallas, USA
[2] META AI

ABSTRACT

A keyword spotting (KWS) engine continuously running on the device is exposed to various speech signals that are usually unseen beforehand. It is a challenging problem to build a small-footprint and high-performing KWS model with robustness under different acoustic environments. In this paper, we explore how to effectively apply adversarial examples to improve KWS robustness. We propose datasource-aware disentangled learning with adversarial examples to reduce the mismatch between the original and adversarial data as well as the mismatch across original training datasources. The KWS model architecture is based on depth-wise separable convolution and a simple attention module. Experimental results demonstrate that the proposed learning strategy improves false reject rate by 40.31% at 1% false accept rate on the internal dataset, compared to the strongest baseline without adversarial examples. Our best-performing system achieves 98.06% accuracy on the Google Speech Commands V1 dataset.

Index Terms: small-footprint keyword spotting, simple attention module, disentangled learning, adversarial examples, depth-wise separable convolution

1. INTRODUCTION

With the proliferation of voice assistant devices, the development of efficient and accurate keyword spotting (KWS) systems has attracted much attention in the literature. These devices rely heavily on an on-device KWS that correctly 'triggers' the system to send audio to cloud for interpretation. Due to constrained hardware resources, on-device KWS systems need to achieve high performance in various acoustic environments with a small memory footprint and high computation efficiency. Advances in this area have a considerable influence on the perceived user experience of the device, as a failure to wake up on a trigger attempt is frustrating, and a false wake on ambient noise can be considered an infraction of user privacy.

Conventional approaches to KWS are based on large vocabulary continuous speech recognition (LVCSR), targeting efficient keyword search from the lattices [1, 2]. However, the LVCSR-based technique often consumes high computational resources and is inapplicable to on-device KWS systems. Hidden Markov Models (HMM) have been a commonly used technique for building small-footprint and low-latency KWS systems, which take either keyword [3, 4] or monophone [5, 6] as modeling units. More recently, deep neural networks (DNNs) have been adopted to predict word units in the keyword for each frame [7]. Convolutional neural networks (CNNs) have seen great success in small-footprint tasks due to their effective representation of time-frequency structure and low memory bandwidth requirements [8] [9] [10].

A KWS system is continuously exposed to various audio signals that are mostly unseen beforehand. Due to the high variability of the acoustic environment in the aspects of SNRs, noise types, and accents, it is a challenging problem to build a robust KWS system that accurately wakes up anytime when the keyword is spoken but reliably suppresses most of the incoming negative audio. Adversarial examples as a free resource have been widely applied to improve model robustness in different tasks by attacking model vulnerability [11] [12] [13]. They are crafted by adding imperceptible perturbations to mislead a well-trained neural network model [14]. Researchers have found that models trained with adversarial examples exhibit unexpected benefits, such as meaningful feature representations that align better with salient data characteristics [15] as well as enhanced robustness to corruptions concentrated in the high-frequency domain [16]. To better leverage adversarial examples for training, Xie et al. have proposed to use an auxiliary batchnorm (BN) specifically for the adversary [17]. Hence the statistics of the original and adversarial data distributions could be more accurately estimated for effective modeling.

In this work, we focus on how to effectively apply the generated adversarial examples to improve the robustness of a small-footprint KWS system. While [18] has explored to use adversarial examples as augmentation data to improve an attention-based KWS system, our interest concentrates on reducing the mismatch between the original training data and adversarial examples during training. Inspired by the effectiveness of the auxiliary BN approach in [17], we propose datasource-aware disentangled adversarial training. Specifically, we design a different auxiliary BN for each type of datasource in the training data and the corresponding adversarial attackers, such that the complementarity across different datasources could be fully exploited. The model architecture of our KWS system is built from the bottleneck residual block [19] that hinges on depth-wise separable convolutions [20]. We extend the residual block by injecting a parameter-free simple attention model (SimAM) [21]. SimAM leverages a 3-dimensional attention map to refine the intermediate feature map in a CNN layer so as to increase model capacity effectively with negligible computation cost. The experimental results have shown the effectiveness of the proposed training approach with adversarial examples. On the internal dataset, we have improved false reject rate by 40.31% at 1% false accept rate, compared to the strongest baseline without using adversarial examples. On the Google Speech Commands V1 dataset, our best KWS system has achieved 98.06% accuracy.

[*] Work performed while the author was an intern at META.

2. MODEL ARCHITECTURE

2.1. Bottleneck Residual Block

Our KWS model adopts the bottleneck residual block in the MobileNetV2 architecture [19] that is tailored to efficient and effective modeling on mobile devices. The bottleneck residual block hinges on the depth-wise separable convolution with the support of the linear bottleneck and inverted residual connection.

The structure of a bottleneck residual block is presented in Fig. 1. Each block is parameterized by its expansion factor t, the number of output channels n, and the stride s. Given an input of shape $h \times w \times m$, the initial pointwise convolution expands the input from m channels to tm channels followed by a 3×3 depth-wise convolution of stride s as well as a linear bottleneck transformation.

The base model in this work is `MN7-45`, which is a variant of the MobileNetV2 architecture optimized for the KWS task. The parameters of `MN7-45` are defined in Table 1. It consists of an initial standard convolution with 45 filters followed by 7 bottleneck residual blocks, global average pooling and a final output layer. Compared to MobileNetV2, `MN7-45` has fewer bottleneck residual blocks but with a larger width in the initial layers, which we found to be more effective for the KWS task. Given a 1s audio input, the number of FLOPS is about 26M.

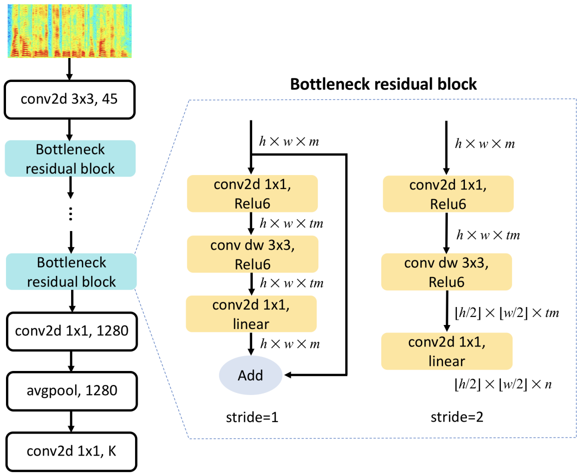

Fig. 1: KWS model architecture (on the left) based on bottleneck residual layers (on the right) that take the input size $h \times w \times m$ and generate output size $\frac{h}{s} \times \frac{w}{s} \times n$.

2.2. Simple Attention Module

Plug-and-play attention modules [22, 23, 24] as an effective component can refine intermediate feature maps within a CNN block to boost the model capacity. The parameter-free simple attention module (SimAM) has shown flexibility and effectiveness in improving the learning capacity of CNNs with negligible computation cost [21]. It infers 3-D attention weights for the feature map in a convolution layer by optimizing an energy function to capture the importance of each neuron. Specifically, the minimal energy of a neuron x in an input feature map $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$ is expressed as:

$$e_x^* = \frac{4(\hat{\sigma}^2 + \lambda)}{(x - \hat{u})^2 + 2\hat{\sigma}^2 + 2\lambda}, \tag{1}$$

Table 1: KWS model architecture `MN7-45` with each layer described by the expansion factor t, the number of output channels n and stride s.

Layer	t	MN7-45		
		n	s	Params
conv2d 3x3	-	45	2	405
bottleneck	6	45	1	26.7K
bottleneck	6	45	2	26.7K
bottleneck	6	45	2	26.7K
bottleneck	6	45	2	26.7K
bottleneck	6	45	1	26.7K
bottleneck	6	45	2	26.7K
bottleneck	6	45	1	26.7K
conv2d 1x1	-	1280	1	57.6K
avgpool	-	1280	-	-
conv2d 1x1	-	2	-	2560

where $\hat{\mu} = \frac{1}{H \times W} \sum_{i=1}^{H \times W} x_i$, $\hat{\sigma}^2 = \frac{1}{H \times W} \sum_{i=1}^{H \times W} (x_i - \hat{\mu})^2$, and λ is a hyper parameter. The statistics μ and σ are shared across all neurons within a channel, significantly reduces computation cost. As neuroscience study indicates an inverse correlation between the energy of e_x^* and the importance of each neuron x [25], the refinement of a feature map can be formulated as,

$$\hat{\mathbf{x}} = \sigma(\frac{1}{\mathbf{E}}) \otimes \mathbf{x}, \tag{2}$$

where $\mathbf{E}$ groups all energy values of e_x^*, and $\sigma(\cdot)$ denotes the sigmoid function. In this work, we plug a SimAM after the depthwise-wise convolution in each residual block of the base model `MN7-45`.

3. DISENTANGLED ADVERSARIAL TRAINING

3.1. Adversarial Examples

Adversarial examples are generated by adding imperceptible but malicious perturbations to the original data, such that the well-trained neural network can be misled to make an incorrect prediction [26]. This work uses the multi-step attacker based on Projected Gradient Descent (PGD) for adversarial data generation [12]. Given an input training sample $\mathbf{x} \in \mathbb{D}$ with the corresponding ground-truth label y, a strong adversary is generated in an iterative manner,

$$\mathbf{x}_{t+1}^{adv} = \Pi_{\mathbf{x}+\mathbb{S}}(\mathbf{x}_t^{adv} + \epsilon \, \mathrm{sgn}(\nabla_{\mathbf{x}} \mathrm{L}(\theta, \mathbf{x}, y))), \tag{3}$$

where Π stands for a projection operator, $\mathbb{S}$ denotes the allowed perturbation space, ϵ is the step size, $\mathrm{L}(\cdot,\cdot,\cdot)$ is the loss function, and θ represents the model parameters. In our work, we use eight steps to generate an adversary. We treat the adversarial examples $\mathbf{x}^{\mathbf{adv}}$ from Eq. 3 as augmented data, and mix them with the original data for training, i.e.,

$$\arg\min_{\theta} [\mathbb{E}_{(\mathbf{x},y)\sim\mathbb{D}}(\mathrm{L}(\theta, \mathbf{x}, y)) + \mathrm{L}(\theta, \mathbf{x}^{\mathbf{adv}}, y)]. \tag{4}$$

3.2. Disentangling via An Auxiliary BN

Previous research work on adversarial attacks has found that training with adversarial examples could lead to label leaking, i.e., the neural network overfits to the specialized adversary distribution, resulting in degraded model performance [11] [26]. To better leverage the regularization power of adversarial data, Xie et al. proposed disentangled training via an auxiliary batchnorm (BN) to decouple the batch statistics between original and adversarial data in normalization layers during network training [17], assuming that the adversarial examples and original data come from different underlying distributions. At each training step, we maintain two BNs, i.e., one main BN and one auxiliary BN, respectively, for the original mini-batch and the corresponding adversarial data, while the rest of network parameters are jointly optimized for both data samples. At the evaluation stage, we keep only the main BN by ignoring the auxiliary one.

3.3. Fine-grained Disentangled Adversarial Training

Adversarial example generation can be easily generalized to a fine-grained version. Instead of generating one adversarial example per training sample, we craft multiple adversaries at different perturbation levels (ϵ of Eq. 3 in a range of $[0.1, 0.4]$) to capture a broader picture of training data statistics. Following basic disentangled learning in Section 3.2, we maintain one main BN for the original training data and a different auxiliary BN for the adversary at each perturbation level.

3.4. Datasource-aware Disentangled Adversarial Training

We further extend the disentangling concept beyond the adversarial examples. The training data for KWS modeling usually has a multi-augmentation composition, including clean audio, audio augmentation with background noise and speaking speed, as well as spectrum distortion with SpecAugment [27]. Each augmentation type could be considered as one datasource. Our premise is that the distribution distinction exists not only between the adversary and original training data but also among the augmentation datasources within the original training data. The datasource-aware disentangled adversarial training hence aims to reduce all the possible mismatches. As illustrated in Fig. 2, we use one main BN for the clean data DS_1 and apply auxiliary BNs respectively for each augmentation datasource DS_n $(n > 1)$ as well as for the corresponding adversarial examples Adv_n $(n \geq 1)$. As described in Section 3.2, we keep the main BN at the evaluation stage by removing all the auxiliary BNs, assuming that the testing data is more likely to follow the clean data distribution than the artificial augmentation distribution. The ablation study presented in Section 5.1 investigates the effectiveness of datasource-aware disentangled learning.

4. EXPERIMENTS

4.1. Data Description

Internal Dataset The internal aggregated and de-identified multi-keyword dataset used in the experiments includes speech samples of four keywords, i. e., "*take a picture*", "*volume down*", "*volume up*", and "*play music*", collected through crowd-sourced workers. The training set contains 130K positive samples, about 32K in each keyword category and 100K negative samples. The test set has 32K positive samples, about 8K for each keyword and 10K negative samples. 6 English accents are involved, including the United States (US), Australia (AU), Canada (CA), New Zealand (NZ), Britain (GB), and Latin America (LA). Training data includes the US accent only, and the other accents are used for testing.

Google Speech Commands Dataset V1 The dataset consists of 1s audio snippets recorded at a sample rate 16kHz in natural environments [28], including $64,727$ samples of 30 different words from $1,881$ speakers. The KWS model is trained to recognize 11 classes: 10 words "*up*", "*down*", "*left*", "*right*", "*yes*", "*no*", "*on*", "*off*", "*go*", "*stop*" as positive samples and the "unknown" category that includes the remaining words and background noise. We split the dataset into training, validation, and testing sets at the ratio of 8:1:1 by following the setting in [28] [29].

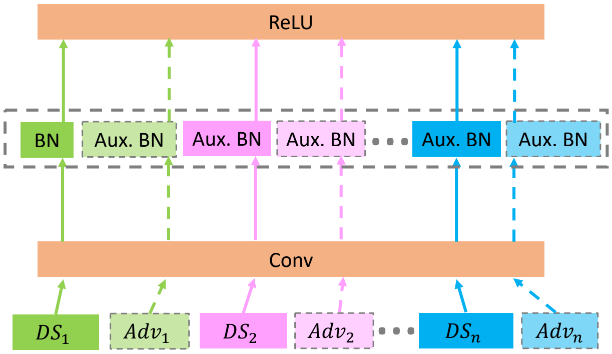

Fig. 2: Information flow in the Datasource-aware disentangled adversarial training architecture (Boxes with solid lines represent each data source, and dashed boxes are the corresponding adversaries of each source).

4.2. Experimental Setup

In the experiments, we consider two popular augmentation baselines: NoiseAugment and SpecAugment [27], in addition to the vanilla training using the clean data. The training data is distorted with various background noises (speech and music) at SNR sampled from [0dB, +20dB]. The testing data is also distorted at SNR of 10dB and 20dB similarly but with different noise types. We further use SpecAugment on top of the noise-augmented training data. Training with adversarial examples (see Section 3) is applied to the 3-datasource training data, including clean, noisy, and SpecAugmented datasources. Specifically, we compare the performance of adversarial training without disentangled learning (AT) in Eq. 4 [18], disentangled adversarial training (DAT), fine-grained disentangled adversarial training (FG_DAT), and datasource-aware disentangled adversarial training (DA_DAT). We use MN7-45 (see Section 2.1) as the base model based on which we also investigate the effectiveness of SimAM for KWS modeling.

We extract acoustic features using 40-dim log Mel-filterbank energies computed over a 25ms window every 10ms. In model training, we use an input window of 900ms (90 frames) for the internal dataset and 1s (100 frames) for the Google speech commands dataset with the output of the corresponding keyword classes. The model inference is performed in a sliding window manner with a shift of 10 frames in all experiments. We use the Adam optimizer with a learning rate of 0.005 and cosine annealing learning rate decay. We train the models for 15 epochs. We perform PGD attackers with perturbation strength ϵ in a range of $[0.1, 0.4]$. We find that $\epsilon = 0.2$ achieves the best performance on the internal dataset and $\epsilon = 0.1$ performs the best on the Google dataset. The hyper-parameter λ of SimAM in Eq. 1 is set to 0.0001. False reject rate (FRR) and false accept rate (FAR) are used for internal data evaluation. We present experimental results by plotting detection error trade-off (DET) curves, where the x-axis and y-axis represent FAR and FRR, respectively. Following the setup in the previous work [28] [29], we measure top-1 classification accuracy for the evaluation of the Google dataset.

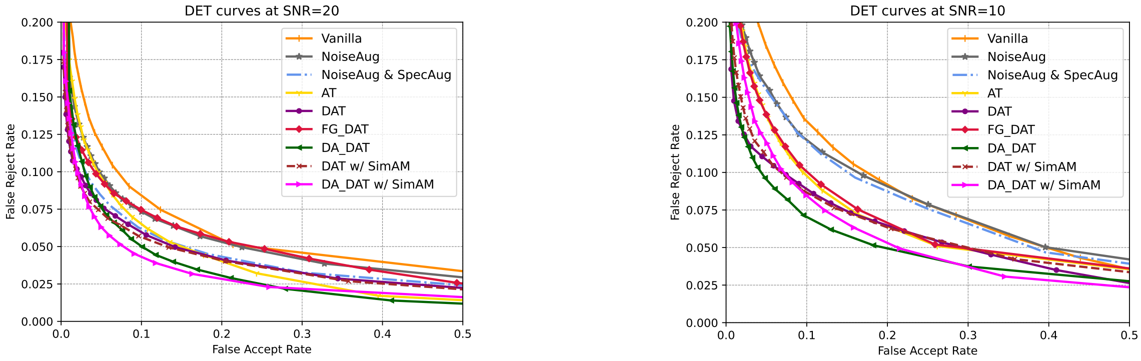

Fig. 3: DET curves of different approaches evaluated on testing data at SNR of 20dB (left) and 10dB (right).

5. RESULTS AND DISCUSSION

5.1. Results on Internal Data

Fig. 3 and Table 2 summarize the results of different approaches on testing data at different SNRs. We can observe that both NoiseAugment and SpecAugment consistently outperform the vanilla training on both distorted datasets. Adversarial examples with different training strategies could further improve performance over the strongest baseline (NoiseAug+SpecAug). This performance boost is especially prominent on highly noisy data. For example, DA_DAT reduces FRR by 20.7% on the SNR $=$ 20dB data and by 40.3% on the SNR $=$ 10dB data at 1% FAR, compared to the baseline using NoiseAug+SpecAug. The results suggest the effectiveness of the adversarial examples for strengthening the generalizability of KWS models in a noisy environment.

Table 2: False reject rate (FRR) of different approaches at 1% false accept rate (FAR).

Methods	SNR $=$ 20dB		SNR $=$ 10dB	
	FRR↓	**AUC**↑	**FRR**↓	**AUC**↑
Vanilla	9.03%	0.956	13.59%	0.939
NoiseAug	7.33%	0.959	12.55%	0.942
NoiseAug+SpecAug	6.28%	0.968	11.98%	0.944
AT	6.20%	0.967	10.04%	0.949
DAT	5.77%	0.970	8.59%	0.960
FG_DAT	7.47%	0.963	10.48%	0.951
DA_DAT	4.98%	0.975	**7.15%**	**0.964**
DAT+SimAM	5.70%	0.971	8.77%	0.956
DA_DAT+SimAM	**4.52%**	**0.976**	8.47%	0.961

Disentangled adversarial training Table 2 shows that DAT has exhibited substantial improvement over AT. We can see an FRR improvement of 7.23% and 13.94%, respectively, on the SNR $=$ 20dB and SNR $=$ 10dB data. This observation has corroborated that the data distribution mismatch does exist between the original data and the adversary, and that disentangled learning with an auxiliary BN is beneficial for bridging such a mismatch.

Datasource-aware disentangled adversarial training We further compare the extensions of disentangled learning, i.e., FG_DAT and DA_DAT, against DAT. It is interesting to observe that DA_DAT surpasses DAT on FRR by 13.69% and 16.76%, respectively on the SNR $=$ 20dB and SNR $=$ 10dB data, while FG_DAT performs worse than even AT. The performance gain from DA_DAT indicates that the datasource-aware approach allows models to learn rich cross-datasource representations and hence increases model robustness against speech distortions. In addition, the auxiliary BN method has demonstrated its effectiveness as a general adaptation technique across datasources. The inferior performance of FG_DAT implies that one perturbation strength is often sufficient for effective attacks of one dataset while multiple perturbation strengths could bring confusion to modeling and degrades model performance.

Simple Attention Module We update the structure of the base model MN7-45 by adding a SimAM (see Sec. 2.2) after the depthwise convolution in each residual block. The updated model is trained using DAT and DA_DAT, the most effective training strategies. It has shown a marginal improvement with DA_DAT on the SNR $=$ 20dB data.

5.2. Results on Google Speech Commands Data

Table 3 presents results on the Google Speech Commands data: the top-1 accuracy associated with the relative decrease in classification error rate (CER_RD). Similar to Section 5.1, FG_DAT obtains an inferior performance. Therefore we omit its results for simplicity. We can observe that adversarial examples help boost KWS performance. Specifically, the base model trained with DA_DAT achieves an accuracy of 97.81% with 18.28% error rate reduction compared to the baseline. Our best-performing system based on the SimAM module and trained with DA_DAT achieves an accuracy of 98.06%.

Table 3: Evaluation results on Google Speech Commands data.

Methods	**Top-1 Acc.**↑	**CER_RD**↑
NoiseAug+SpecAug	97.32%	-
AT	97.64%	11.94%
DAT	97.74%	15.67%
DA_DAT	97.81%	18.28%
DA_DAT+SimAM	**98.06%**	**27.61%**

6. CONCLUSIONS

This paper explored how to effectively apply adversarial examples for KWS modeling. We proposed datasource-aware disentangled training with adversarial examples through multiple auxiliary BNs. Experimental results on both internal dataset and Google Speech Commands dataset have demonstrated that adversarial examples as a free and infinite data resource could effectively boost KWS performance and that the proposed training strategy exerts effectiveness to a large extent.

7. REFERENCES

[1] D. R. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection," in *Eighth Annual Conference of the international speech communication association*, 2007.

[2] J. Wilpon, L. Rabiner, C.-H. Lee, and E. Goldman, "Automatic recognition of keywords in unconstrained speech using hidden markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 11, pp. 1870–1878, 1990.

[3] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden markov modeling for speaker-independent word spotting," in *International Conference on Acoustics, Speech, and Signal Processing,*. IEEE, 1989, pp. 627–630.

[4] R. C. Rose and D. B. Paul, "A hidden markov model based keyword recognition system," in *International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1990, pp. 129–132.

[5] M. Wu, S. Panchapagesan, M. Sun, J. Gu, R. Thomas, S. N. P. Vitaladevuni, B. Hoffmeister, and A. Mandal, "Monophone-based background modeling for two-stage on-device wake word detection," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5494–5498.

[6] M. Sun, D. Snyder, Y. Gao, V. K. Nagaraja, M. Rodehorst, S. Panchapagesan, N. Strom, S. Matsoukas, and S. Vitaladevuni, "Compressed time delay neural network for small-footprint keyword spotting." in *Interspeech*, 2017, pp. 3607–3611.

[7] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4087–4091.

[8] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech 2015*, 2015, pp. 1478–1482.

[9] 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.

[10] 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.

[11] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[12] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[13] T. Pang, X. Yang, Y. Dong, K. Xu, J. Zhu, and H. Su, "Boosting adversarial training with hypersphere embedding," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7779–7792, 2020.

[14] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *2018 IEEE security and privacy workshops (SPW)*. IEEE, 2018, pp. 1–7.

[15] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "There is no free lunch in adversarial robustness (but there are unexpected benefits)," *arXiv:1805.12152*, 2018.

[16] D. Yin, R. Gontijo Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A fourier perspective on model robustness in computer vision," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 819–828.

[18] X. Wang, S. Sun, C. Shan, J. Hou, L. Xie, S. Li, and X. Lei, "Adversarial examples for improving end-to-end attention-based small-footprint keyword spotting," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6366–6370.

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[20] L. Sifre and S. Mallat, "Rigid-motion scattering for texture classification," *arXiv preprint arXiv:1403.1687*, 2014.

[21] L. Yang, R.-Y. Zhang, L. Li, and X. Xie, "Simam: A simple, parameter-free attention module for convolutional neural networks," in *International conference on machine learning*. PMLR, 2021, pp. 11 863–11 874.

[22] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[23] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[24] Z. Yang, L. Zhu, Y. Wu, and Y. Yang, "Gated channel transformation for visual recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 794–11 803.

[25] B. S. Webb, N. T. Dhruv, S. G. Solomon, C. Tailby, and P. Lennie, "Early and late mechanisms of surround suppression in striate cortex of macaque," *Journal of Neuroscience*, vol. 25, no. 50, pp. 11 666–11 675, 2005.

[26] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[27] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

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

[29] O. Rybakov, N. Kononenko, N. Subrahmanya, M. Visontai, and S. Laurenzo, "Streaming keyword spotting on mobile devices," *arXiv preprint arXiv:2005.06720*, 2020.