DOMAIN GENERALIZATION WITH RELAXED INSTANCE FREQUENCY-WISE NORMALIZATION IN 2D NEURAL AUDIO PROCESSING

Anonymous authors
Paper under double-blind review

ABSTRACT

While using two-dimensional convolutional neural networks (2D-CNNs) in image processing, it is possible to manipulate domain information using channel statistics, and instance normalization has been a promising way to get domain-invariant features. Although 2D image features represent spatial information, 2D audio features like log-Mel spectrogram represent two different temporal and spectral information. Unlike image processing, we analyze that domain-relevant information in the audio feature is dominant in frequency statistics rather than channel statistics. Motivated by our analysis, we introduce RFN, a plug-and-play, explicit normalization module along the frequency axis, eliminating instance-specific domain discrepancy in the audio feature while *relaxing* undesirable loss of useful discriminative information. Empirically, simply adding RFN to networks shows clear margins compared to previous domain generalization approaches on acoustic scene classification, keyword spotting, and speaker verification tasks and yields improved robustness to audio-device, speaker-ID, or genre.

1 INTRODUCTION

To minimize the expectation of loss functions, *risk*, over the intractable data space, it has been commonly used to minimize the empirical risk (expectation over training dataset) with i.i.d assumption in machine learning (Vapnik, 1991). However, the assumption does not always hold, and deep neural networks (DNNs) have difficulty in being generalized to unseen domains, which can cause poor results in real-world scenarios. Hence, in most fields, including computer vision, audio processing, and natural language processing, *domain generalization* (DG) has been an essential research topic.

LeCun et al. (1989) and Krizhevsky et al. (2012) introduced two-dimensional convolutional neural networks (2D-CNNs) in the image recognition task. Since then, 2D-CNN architecture has been widely employed beyond the image field. Paying attention to that domain-relevant information is reflected in the channel statistics of the convolutional features of images, several works (Nam & Kim, 2018; Pan et al., 2018; Choi et al., 2020; Bronskill et al., 2020) exploited Instance Normalization (IN) (Ulyanov et al., 2016; 2017) to eliminate instance-specific domain discrepancy. This approach is promising for achieving domain-invariant representations. Here, we raise a question: *Does this approach make sense in other fields, especially in audio?*

In audio, temporal and frequency dimensions convey important information. Therefore, it has been *de facto* to represent audio signals with 2D representation such as log-Mel spectrogram and Mel-Frequency Cepstral Coefficients (MFCCs). Like the image field, taking those 2D representations as inputs, 2D-CNNs have also been adopted in diverse audio tasks, e.g., audio scene classification, speech recognition, and speaker recognition. However, whereas 2D convolution is operated along spatial dimensions in the image field, it convolves the frequency and temporal information in the audio field. Hence, dissimilar with images, domain information may not be mainly distributed in channel statistics in audio. Although various works have addressed better usage of 2D-CNNs in the audio field (Hershey et al., 2017; Park et al., 2019; Phaye et al., 2019; McDonnell & Gao, 2020; Palanisamy et al., 2020; Chang et al., 2021; Kim et al., 2021a), there has been less effort to focus on dimensional characteristics of 2D audio representation for DG in neural architectures.

This work focuses on domain generalization by explicit normalization, manipulating statistics in 2D audio features. In particular, we analyze the relationship between the domain and the statistics of each feature dimension by estimating mutual information, and demonstrate that the *frequency feature* (including spectrum and cepstrum) dimension carries more domain-relevant information than channel and temporal dimensions. We also find that the frequency dimension contains a meaningful representation of class discriminative information. Motivated by the analysis, we introduce a plug-and-play DG module named Relaxed instance Frequency-wise Normalization (RFN). RFN removes domain information through normalization along frequency dimension at the input and hidden layers of networks. We also introduce the term *relaxation* that can effectively control the degree of normalization to prevent the loss of useful discriminative information. We provide experimental analyses of RFN on two classification tasks and one verification task for environmental sound and human voice addressing multiple domains: acoustic scene classification (ASC), keyword spotting (KWS), and speaker verification (SV). We empirically observe that the performing DG on the frequency dimension is essential for audio features in 2D-CNNs.

We make the following contributions: (1) In 2D audio processing, we analyze that domain information is highly correlated to frequency statistics. Our analysis show that it is possible and better to explicitly manipulate frequency statistics rather than channel statistics to eliminate domain discrepancy per instance; (2) We propose a novel domain generalization method, Relaxed instance Frequency-wise Normalization (RFN) to manipulate instance frequency statistics. To the best of our knowledge, this is the first work to solve domain generalization based on the different characteristics of channel and frequency statistics in audio features; (3) Through extensive experiments, we demonstrate that our RFN can discard unnecessary domain information while avoiding undesirable loss of discriminative information. Simply adding our RFN to networks shows clear margins compared to other DG methods in various audio tasks and yields better device, speaker ID, and genre robustness.

2 Characteristics of Audio in 2D CNNs

We consider characteristics of 2D audio representation in $\mathbb{R}^{F\times T}$ as an input of 2D-CNNs where F and T stands for frequency and time axes, respectively. Using a mini-batch axis N, we can represent an element of activation, $\boldsymbol{x}$, with a 4D-index of $i = (i_N, i_C, i_F, i_T)$, where i_C is the channel-index. Motivated by that the instance statistics along a specific dimension can represent domain-relevant information (Huang & Belongie, 2017; Nam & Kim, 2018; Zhou et al., 2021), we consider the domain information of each feature dimension in 2D-CNNs.

2.1 Preliminaries

Estimation of Mutual Information. Direct computing of mutual information (MI) is usually intractable for continuous random variables. Following the literature (Wang et al., 2021b), we add an auxiliary classifier with parameter ψ on top of the statistics along a specific dimension of activation $\boldsymbol{x}$, and train the classifier to correctly classify d or y, one-hot ground-truth domain or task label, respectively. More precisely, Wang et al. (2021b) approximated MI, $I(\boldsymbol{x}, y) = H(y) - E_{p(\boldsymbol{x},y)}[-\log p(y|\boldsymbol{x})]$ by approximating the expectations of true $p(y|\boldsymbol{x})$ by the expectations of $q_\psi(y|\boldsymbol{x})$ over a training set of size M. The resulting estimation is $I(\boldsymbol{x}, y) = H(y) - \frac{1}{M}\sum_{i=1}^{M} -\log q_\psi(y_i|\boldsymbol{x}_i)$. They simply use the test accuracy of $q_(\psi)(y|\boldsymbol{x})$ as an estimation of $I(\boldsymbol{x}, y)$ which is highly correlated to $\frac{1}{M}\sum_{i=1}^{M} -\log q_\psi(y_i|\boldsymbol{x}_i)$, the cross-entropy loss (same of d).

Notations. We generate instance statistics, mean and standard deviation (std), along a specific dimension for activation $\boldsymbol{x}$ of the 4D-index i. For instance, we denote instance frequency-wise statistics of $\boldsymbol{x}_{i_N}$ as $\boldsymbol{s}^{(F)}$ (omit index i_N for clarity) which is a concatenation of mean and std along F-axis, $\boldsymbol{s}^{(F)} = \text{concat}(\boldsymbol{\mu}^{(F)}, \boldsymbol{\sigma}^{(F)})$, where $\boldsymbol{\mu}^{(F)}, \boldsymbol{\sigma}^{(F)} \in \mathbb{R}^F$. In image, there are spatial axes, H (height) and W (width) instead of F and T, respectively.

2.2 Analysis of Instance Statistics of 2D Audio Features

For the MI analysis in the image, we employ ResNet18 (He et al., 2016) trained on a multi-domain image dataset, PACS (Li et al., 2017), which has seven common categories for y with four domains (photo, art-painting, cartoon, and sketch) for d. In audio, we use BC-ResNet-Mod-1 (Kim et al.,

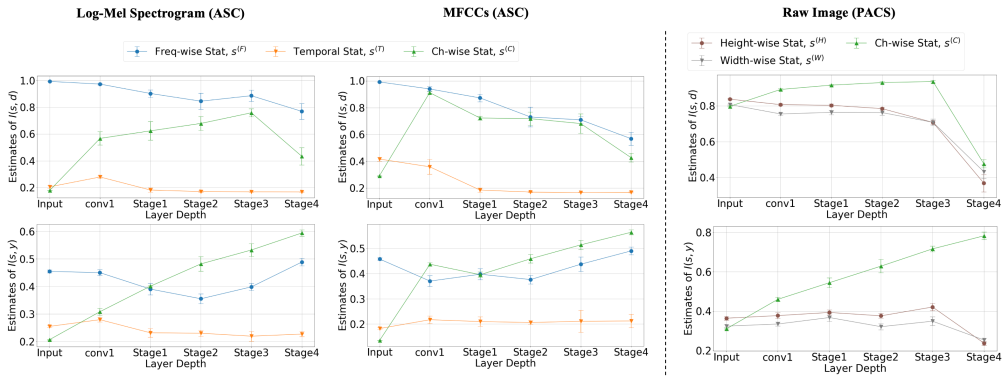

Figure 1: **Estimates of mutual information** between dimension-wise statistics of intermediate hidden feature and the domain label, $I(\boldsymbol{s}, d)$ (top), or the class label, $I(\boldsymbol{s}, y)$ (bottom). We use log-Mel spectrogram (left) and MFCC (middle) on the audio scene classification task and use raw RGB image (right) for the PACS dataset (Average over 5 seeds; error bar stands for standard deviation).

2021a;b) suitable for DCASE 2021, multi-device acoustic scene classification (ASC) task (Heittola et al., 2020) which has ten acoustic scenes (e.g., airport, shopping mall, metro station) for y with six different train recording devices (domains) for d. (We denote BC-ResNet-Mod by BC-ResNet for clarity.) The ASC validation set contains devices which are unseen during training, and we did six-way classification for seen devices (3 real and 3 simulated devices) to get estimation of $I(\boldsymbol{s}, d)$. For the ASC task, we exploit two representative 2D audio features: log-Mel spectrogram and MFCCs. We estimate MI after each *stage* which is a sequence of convolutional blocks whose activations have the same width.

The first row of Figure 1 plots the estimates of MI of each dimension varying stages for domain labels. In PACS (right), the estimate of $I(\boldsymbol{s}^{(C)}, d)$ is higher than others in every stage and increases until stage 3. Whereas the estimated MI of width or height-wise ones gradually decreases as layers go deep. The observation correlates well with the common belief that channel-wise instance statistics can represent *style* (domain) in image-level 2D-CNNs (Huang & Belongie, 2017). Contrarily, in audio (left and middle), MI of frequency-wise statistics (Freq-wise Stat) is a par or superior to that of channel statistics while MI of temporal statistics is inferior on all stages.

The second row of Figure 1 shows the estimates of MI for task label y. In the image, channel statistics again yields higher values than the width or height-wise ones. However, we observe that frequency statistics is also highly correlated to class label y in the audio domain. Therefore, we can infer that it is essential to suppress unnecessary domain information and preserve the task-relevant information in the frequency dimension for domain generalization in audio.

t-SNE Visualization. We further analyze the 2D t-SNE visualization (Van der Maaten & Hinton, 2008) to compare $\boldsymbol{s}^{(F)}$ and $\boldsymbol{s}^{(C)}$. In Figure 2 the features are separated better by device-IDs (A to S3) with frequency-wise mean and std, $\boldsymbol{s}^{(F)}$, compared to that of the channel, $\boldsymbol{s}^{(C)}$. The observation support the analysis of MI estimation. See Figure A5 for more plots.

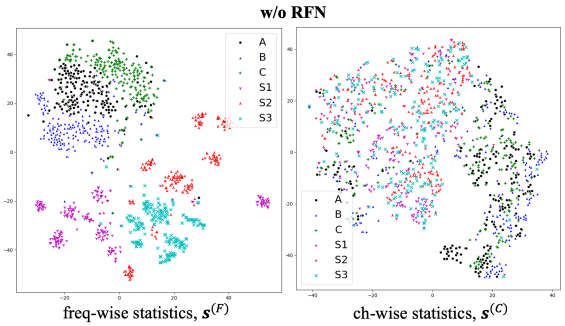

Figure 2: **2D t-SNE visualiations** using activation of stage 1 in BC-ResNet-1.

3 Relaxed Instance Frequency-wise Normalization

Instance Normalization (IN) (Ulyanov et al., 2016) has been one of the promising ways to get domain-invariant features for various domain generalization (Nam & Kim, 2018; Pan et al., 2018; Choi et al., 2020) and transfer approaches (Huang & Belongie, 2017) in image processing. However, in Section 2, the MI analysis showed that domain information is highly contained in the frequency

Algorithm 1 RFN, applied to activation $\boldsymbol{x}$ over a mini-batch.

Input: Activation $\boldsymbol{x} \in \mathbb{R}^{N \times C \times F \times T}$ over a mini-batch. Relaxation $\lambda \in [0, 1]$.
Output: $\boldsymbol{y} = \mathit{RFN}(\boldsymbol{x})$

$\mathrm{E}_{\mathit{IFN}}[\boldsymbol{x}] \leftarrow \frac{1}{C \cdot T} \sum_c^C \sum_t^T \boldsymbol{x}_{:,c,:,t}$
$\mathrm{Var}_{\mathit{IFN}}[\boldsymbol{x}] \leftarrow \frac{1}{C \cdot T} \sum_c^C \sum_t^T (\boldsymbol{x}_{:,c,:,t} - \mathrm{E}_{\mathit{IFN}}[\boldsymbol{x}]) \odot (\boldsymbol{x}_{:,c,:,t} - \mathrm{E}_{\mathit{IFN}}[\boldsymbol{x}])$
$\mathit{IFN}(\boldsymbol{x})_{:,c,:,t} \leftarrow \frac{\boldsymbol{x}_{:,c,:,t} - \mathrm{E}_{\mathit{IFN}}[\boldsymbol{x}]}{\sqrt{\mathrm{Var}_{\mathit{IFN}}[\boldsymbol{x}] - \epsilon}}$ ▷ IFN
$\mathrm{E}_{\mathit{LN}}[\boldsymbol{x}] \leftarrow \frac{1}{C \cdot F \cdot T} \sum_c^C \sum_f^F \sum_t^T \boldsymbol{x}_{:,c,f,t}$
$\mathrm{Var}_{\mathit{LN}}[\boldsymbol{x}] \leftarrow \frac{1}{C \cdot F \cdot T} \sum_c^C \sum_f^F \sum_t^T (\boldsymbol{x}_{:,c,f,t} - \mathrm{E}_{\mathit{LN}}[\boldsymbol{x}]) \odot (\boldsymbol{x}_{:,c,f,t} - \mathrm{E}_{\mathit{LN}}[\boldsymbol{x}])$
$\mathit{LN}(\boldsymbol{x})_{:,c,f,t} \leftarrow \frac{\boldsymbol{x}_{:,c,f,t} - \mathrm{E}_{\mathit{LN}}[\boldsymbol{x}]}{\sqrt{\mathrm{Var}_{\mathit{LN}}[\boldsymbol{x}] - \epsilon}}$ ▷ LN
$\boldsymbol{y} \leftarrow \lambda \cdot \mathit{LN}(\boldsymbol{x}) + (1 - \lambda) \cdot \mathit{IFN}(\boldsymbol{x})$ ▷ RFN

dimension in the audio features. Also we observed that the audio features are better grouped by domain label while using frequency statistics compared to that of channel. Hence, we consider using instance frequency-wise normalization (IFN) instead of IN for the domain generalization in audio.

Frequency-wise normalization. Before defining IFN, we start by reviewing the formulation of conventional channel-wise normalization (CN), which includes Batch Normalization (BN) (Ioffe & Szegedy, 2015), IN, and Group Normalization (GN) (Wu & He, 2018), to distinguish them from frequency-wise normalization (FN). We denote an element of a normalized feature as $\hat{x}_i = (x_i - \mu_i)/\sigma_i$ where $\mu_i = \frac{1}{m} \sum_{k \in S_i} x_k$ and $\sigma_i^2 = \frac{1}{m} \sum_{k \in S_i} (x_k - \mu_i)^2 + \epsilon$ are calculated over an index-set S_i whose size $|S_i| = m$. Then. feature normalization methods can be defined by the set S_i. We define CN as a feature normalization, where feature elements having same channel index, i_C, share the mean and variance, e.g., BN is defined by $S_i = \{k | k_C = i_C\} = \{k | k = (:, i_C, :, :)\}$.

Now, we define FN as a feature normalization, where feature elements sharing same frequency feature index, f, are normalized together. Similar to CN, we can get Batch-FN (BFN), IFN, and Group-FN (GFN), and we can define IFN by following S_i.

$$\text{IFN: } S_i = \{k | k_N = i_N, k_F = i_F\}. \tag{1}$$

Relaxed instance frequency-wise normalization. We use IFN to eliminate instance-specific domain discrepancy represented in the frequency distribution. However, we also observe that frequency statistics is highly correlated to both domain information and class-discriminative information in Section 2, and thus, we can expect a possible loss of useful discriminative information from the direct use of IFN. We try to *relax* (alleviate) the effect of IFN by using an additional normalization. We use instance-wise global statistics from $S_i = \{k | k_N = i_N\}$, which is also known as Layer Normalization (LN) (Ba et al., 2016). Using IFN and LN, we introduce a novel domain generalization module, named, Relaxed instance Frequency-wise Normalization (RFN) as following:

$$\mathit{RFN}(\boldsymbol{x}) = \lambda \cdot \mathit{LN}(\boldsymbol{x}) \ + (1 - \lambda) \cdot \mathit{IFN}(\boldsymbol{x}), \tag{2}$$

where $\boldsymbol{x}$ is an input to RFN, and $\lambda \in [0, 1]$ represents the degree of relaxation. Larger λ implies more relaxation. We do not use affine transformation for IFN and additional normalizatoin, LN. The resulting mean and std of $\mathit{RFN}(\boldsymbol{x})$ are $\hat{\mu}_i^{(F)} = \frac{\lambda}{\sigma_i}(\mu_i^{(F)} - \mu_i)$ and $\hat{\sigma}_i^{(F)} = \lambda \cdot \frac{\sigma_i^{(F)}}{\sigma_i} + (1 - \lambda)$, respectively (See Appendix A.2) , where $\mu_i^{(F)}$ and $\sigma_i^{(F)}$ are calculated over $S_i = \{k | k_N = i_N, k_F = i_F\}$, and μ_i and σ_i are statistics over $S_i = \{k | k_N = i_N\}$. The values of $\hat{\mu}_i^{(F)}$ and $\hat{\sigma}_i^{(F)}$ correspond to mean and std of $\mathit{LN}(\boldsymbol{x})$ and that of $\mathit{IFN}(\boldsymbol{x})$ when $\lambda = 1$ and 0, respectively. The key feature of relaxation is adding a normalized feature to IFN, not LN. In this work, we experiment LN with IFN for RFN, but there are other possible design choices like BN or GN instead of LN depending on a task and a model architecture. For more discussion, see Appendix A.3.

We present the RFN transform in algorithm 1 which is simple and easy to implement (example PyTorch-like code in Appendix A.1). RFN can be inserted in the middle of the existing network to erase unnecessary instance-specific domain discrepancy as an additional, plug-and-play module, and it is also expected to avoid undesirable loss of task-relevant information through relaxation.

4 RELATED WORKS

Normalization-based DG. The literature in domain generalization (DG) is vast and out of scope for this work. Hence, we review the most relevant works. There have been various IN-based approaches, e.g., BIN (Nam & Kim, 2018), IBN-Net (Pan et al., 2018), and Meta-BIN (Choi et al., 2020), to get domain (style) invariant features by exploiting the property of image processing that domain information can be captured in channel-wise distribution. They are effective, but IN is a task-agnostic normalization layer that can cause lower discriminative performance (Ulyanov et al., 2017). Therefore, they used IN with BN to alleviate the effect of IN as alternatives to conventional BN in neural architectures, but limited to BN. Bronskill et al. (2020) proposed TaskNorm, combinations of BN with IN or LN, to overcome the mismatch of data distributions in meta-learning scenarios. A plug-and-play module, MixStyle (Zhou et al., 2021) got domain robustness using mixed channel-wise statistics based on Adaptive IN (AdaIN) (Huang & Belongie, 2017), an image style transfer approach. Despite the relation to these approaches, we address that those approaches are not suitable for manipulating domain information in audio features.

Other DG approaches. There have been other branches for DG besides using explicit normalization. We introduce some notable works; (1) Data-augmentation approaches augment inputs or hidden features to increase the diversity of seen domains to cover unseen data space, e.g., general-purpose regularization or adversarial gradient-based methods (Zhang et al., 2018; Volpi et al., 2018; Shankar et al., 2018; Zhou et al., 2020; Xu et al., 2021). They are effective, but still, depending on the size of the training dataset and the number of seen domains. (2) There have been feature disentanglement methods that decompose weights or hidden features into domain-specific and common parts (Peng et al., 2019; Piratla et al., 2020), represented by Common-Specific Decomposition (CSD) (Piratla et al., 2020), which decomposes the last softmax layer into domain-common and domain-specific weights. (3) Moreover, there have been learning methods like meta-learning (learning-to-learn) for DG scenarios (Li et al., 2018; Kang et al., 2020; Zhao et al., 2021). The above categories are conceptually different and can be used together for better generalization capability (Wang et al., 2021a). RFN can also be complementary to and not a replacement for them, as shown in later experiments.

Frequency-wise computations. Frequency-wise computations have been attempted to get robust audio features, and frequency-wise normalization has been more widely studied recently (Chang et al., 2021; Yeung et al., 2021; Stafylakis et al., 2021). For example, Wang et al. (2017) introduced per-channel energy normalization (PCEN), a moving average of frequency-wise energy over time, and Zeghidour et al. (2018) and Park & Yoo (2020) observed that IFN is beneficial for a learnable filterbank. Cepstral mean and variance normalization (CMVN) (Viikki & Laurila, 1998; Prasad & Umesh, 2013) shows that such normalization also works for MFCCs. They are successful to achieve robust features, but are also suffer from information loss due to instance-wise normalization. Also the methods are limited to input preprocessing. The proposed RFN is more flexible that it can be used in the middle of the network, and we introduce additional normalized feature term, relaxation, to successfully compensate undesirable information loss.

5 EXPERIMENTS

5.1 DATASETS AND EXPERIMENTAL SETUP

To verify DG capability of RFN, we experiment with RFN on three different audio tasks composed of two classification tasks, audio scene classification (ASC) and keyword spotting (KWS), and one verification task, speaker verification (SV).

Multi-device audio scene classification. In ASC, it is required to classify an audio segment to one of the given acoustic scene labels. We use the TAU Urban Acoustic Scenes 2020 Mobile development dataset (Heittola et al., 2020), which consists of training and validation data of 13,962 and 2,970 audio segments, respectively. The data are recorded from 12 European cities in 10 different acoustic scenes using three real (A, B, and C) and six simulated devices (S1-S6). The task is challenging due to domain imbalance (device A is dominant) and unseen domains (devices S4-S6 are not used during training). We use the ASC version of BC-ResNets (Kim et al., 2021b), which recently won the task in DCASE2021 (Martín-Morató et al., 2021) and another baseline architecture, CP-ResNet, c=64 (Koutini et al., 2019).

Table 1: **Acoustic Scene Classification.** Top-1 validation accuracy (%) on TAU Urban Acoustic Scenes 2020 Mobile development dataset. (average and standard deviation; averaged over 5 seeds)

Method	#Param	seen						unseen			Overall	Δ
		A	B	C	S1	S2	S3	S4	S5	S6		
BC-ResNet-8	315k	79.6	70.8	74.3	69.8	66.2	72.8	63.6	63.3	59.2	68.9 ± 0.8	+ 0.0
+ Global FreqNorm	315k	80.2	72.2	**76.2**	70.8	67.5	72.6	65.4	66.0	56.2	69.7 ± 0.6	+ 0.8
+ PCEN	315k	75.6	66.7	66.3	69.0	67.0	73.6	68.1	68.3	66.7	69.0 ± 0.7	+ 0.1
+ Mixup	315k	79.9	70.3	72.0	69.8	65.9	70.1	60.5	60.8	56.1	67.3 ± 1.0	- 1.6
+ MixStyle	315k	78.5	70.0	72.0	68.4	65.9	68.3	59.0	59.3	54.6	66.2 ± 0.7	- 2.7
+ BIN	317k	76.9	70.2	71.4	67.3	65.6	69.6	60.4	62.2	57.6	66.8 ± 1.5	- 2.1
+ CSD	317k	77.5	71.4	72.8	68.7	66.8	71.0	65.0	63.5	56.7	68.2 ± 0.4	- 0.7
+ RFN (Ours)	315k	**82.4**	**73.2**	74.5	**75.7**	69.9	**76.9**	**70.5**	**72.4**	**69.5**	**73.9 ± 0.7**	**+ 5.0**
$\lambda = 0$ (IFN)	315k	77.1	71.7	67.8	73.0	**71.1**	74.8	69.8	70.9	67.4	71.5 ± 1.2	+ 2.6
$\lambda = 1$ (LN)	315k	79.8	71.9	72.9	73.6	68.9	70.7	62.2	63.2	57.2	68.9 ± 0.7	+ 0.0
BC-ResNet-1	8.1k	73.3	61.3	**64.9**	61.0	58.3	66.7	51.8	51.3	48.5	59.7 ± 1.3	+ 0.0
+ RFN (Ours)	8.1k	**75.2**	**63.7**	64.0	**62.8**	**61.2**	**68.0**	**58.3**	**63.0**	**57.2**	**63.7 ± 0.9**	**+ 4.0**
CP-ResNet, $c = 64$	897k	78.1	**71.2**	**73.4**	68.3	65.9	68.7	64.8	64.8	58.5	68.2 ± 0.4	+ 0.0
+ RFN (Ours)	897k	**79.3**	70.9	70.8	**71.8**	**72.1**	**74.1**	**69.9**	**68.6**	**66.0**	**71.5 ± 0.3**	**+ 3.3**

Keword spotting. KWS aims to detect and classify predefined keywords. We use Google Speech Commands dataset version 1 (Warden, 2018), which contains 64,727 utterances from 1,881 speakers. Aside from audio device-ID in the ASC task, speaker ID is another important domain, which can characterize audio data. Following the DG settings of Shankar et al. (2018) and Piratla et al. (2020), we experiment with the architecture of Sainath & Parada (2015), *cnn-trad-fpool3*, with varying numbers of training speakers from 50 to a large number of domains, 1,000, and observe performance for unseen speakers. The task is a 12-way classification for ten target keywords (among 30 words) and two additional classes, 'silence' and 'unknown words' (the rest 20 words) (Warden, 2018). We split validation and test speakers using 10% of the total speakers for each. We also experiment another backbone, ResNet15 (Tang & Lin, 2018).

Multi-genre speaker verification. We also try RFN on a verification task, SV, which accepts or rejects a speaker's utterance's claimed identity based on a few enrolled utterances from a target speaker. We experiment on the CN-Celeb dataset (Fan et al., 2020; Li et al., 2020), which has a total of 659,594 utterances from 3,000 speakers recorded in 11 genres, *e.g.*, 'interview,' 'drama,' and 'singing,' from various platforms and devices. We use 'singing' and 'movie' as unseen genres during training following (Li et al., 2020), and the dataset splits 200 speakers as CN-Celeb (E) for validation (Fan et al., 2020). The task is especially challenging due to the severe imbalance between genres in each speaker; more than half of speakers record only one genre. We exploit Fast ResNet-34 (Chung et al., 2020; Heo et al., 2020), a variant of the ResNet with 34 layers, suitable for speaker verification, and train the model with AM-Softmax, which has shown notable improvements over cross-entropy loss (Wang et al., 2018a;b). The results are obtained using cosine-similarity without any score or other normalization.

Baselines. We compare our approach to the following baselines. We use (1) 'Global FreqNorm' to normalize inputs using global frequency-wise statistics from training data as a naive baseline. Next, we experiment (2) Per-Channel Energy Normalization (PCEN) (Wang et al., 2017), which preprocess audio input by moving average of frequency-wise energy along the temporal axis. We use PCEN instead of log-Mel. We compare IN-based approaches, (3) MixStyle (Zhou et al., 2021), mixing channel-wise statistics within mini-batch in the middle of a network, and (4) BIN (Nam & Kim, 2018), a combination of BN and IN, which switches all BNs in a network. We also compare other notable DG approaches, (5) a general regularization approach, Mixup (Zhang et al., 2018), (6) an adversarial gradient-based method, CrossGrad (Shankar et al., 2018), and (7) a decomposition method, CSD (Piratla et al., 2020). We use the official implementations of MixStyle, BIN, Mixup, and CSD (Zhou et al., 2021; Nam & Kim, 2018; Zhang et al., 2018; Piratla et al., 2020).

We set $\lambda = 0.5$ for all RFN as default. See Appendix A.4 for more details of experimental setups.

5.2 Experimental Results

Multi-device acoustic scene classification. The results are shown in Table 1. The vanilla BC-ResNet-8 shows poor device generalization capability. The dominant one, 'A,' shows the highest

Table 2: **Keyword Spotting.** Top-1 test accuracy (%) with varying number of training speakers on Google speech command dataset ver1. (average and standard deviation; averaged over 5 seeds)

Method	50	100	200	1000
cnn-trad-fpool3	72.5 ± 0.3	80.5 ± 0.5	86.9 ± 0.5	92.3 ± 0.4
+ Mixup	70.5 ± 0.5	79.7 ± 0.4	$\mathbf{87.9 \pm 0.4}$	$\mathbf{93.3 \pm 0.4}$
+ CrossGrad	73.7 ± 0.8	81.1 ± 0.5	87.2 ± 0.2	92.6 ± 0.2
+ CSD	73.4 ± 0.5	80.9 ± 0.6	87.5 ± 0.5	92.8 ± 0.3
+ RFN (Ours)	$\mathbf{76.6 \pm 0.6}$	$\mathbf{82.3 \pm 0.6}$	$\mathbf{87.8 \pm 0.6}$	92.4 ± 0.3
$\lambda = 0$ (IFN)	71.1 ± 1.1	77.9 ± 0.7	85.3 ± 0.3	90.9 ± 0.3
$\lambda = 1$ (LN)	72.7 ± 0.8	80.8 ± 0.7	86.5 ± 0.3	92.4 ± 0.2
ResNet15	84.4 ± 1.1	87.9 ± 0.4	91.1 ± 0.2	94.9 ± 0.2
+ RFN (Ours)	$\mathbf{86.3 \pm 1.0}$	$\mathbf{90.2 \pm 0.5}$	$\mathbf{91.8 \pm 0.4}$	$\mathbf{95.6 \pm 0.2}$

Table 3: **Speaker verification.** EER (%) on overall genres and each genre on CN-Celeb. The numbers are average and standard deviation (average over 5 seeds).

Method	seen						unseen		Overall
	drama	entertainment	interview	live	speech	vlog	movie	singing	
Baseline	13.1	14.6	11.4	13.8	5.7	7.6	18.4	28.7	14.5 ± 0.4
Mixup	13.9	14.3	10.7	**12.0**	4.6	**6.3**	**16.8**	28.9	14.1 ± 0.2
MixStyle	14.2	14.2	11.1	13.0	5.5	7.0	19.7	28.1	14.0 ± 0.3
BIN	14.1	15.5	12.2	14.0	5.3	8.4	18.3	29.5	15.0 ± 0.6
RFN (Ours)	**11.9**	**13.5**	**10.1**	12.9	**4.1**	7.2	18.7	**27.4**	$\mathbf{13.4 \pm 0.3}$

performance while the unseen devices' performance is much lower compared to seen devices. Overall, the baselines do not show clear improvements compared to the vanilla BC-ResNet-8 for both seen and unseen domains. PCEN seems to help device generalization by frequency-wise input preprocessing, but it leads to performance degradation for seen devices. We attach five RFNs at the input and after every stage of the networks. Our approach shows significant improvements, relatively more than 10% in top-1 accuracy for unseen domains, while still shows promising results for seen devices. When we directly use IFN by $\lambda = 0$ in RFN, it also shows good generalization capability but lower performance than RFN especially for seen devices. Also, direct use of LN by $\lambda = 0$ does show clear improvement here. (See Appendix A.5 for more experimental results using less seen domain, BC-ResNet-1, or CP-ResNet with c=64.)

Keyword spotting. This task desires robustness over speaker IDs. In Table 2, both CrossGrad and CSD show near 0.5 to 1% improvements for the varying number of training speakers compared to the vanilla cnn-trad-fpool3. Mixup shows improvements with a large number of domains, but it works poorly with a small number of training speakers. Here, we use RFN only at the input as an additional module, considering the shallow architecture of cnn-trad-fpool3. Compared to baselines, RFN especially shows clear margins when the number of training speakers is small. RFN also shows clear improvements in ResNet15. There is no separation like the *stage* in the architecture, thus we simply add RFN at input and after every ResBlock (in total there are seven RFNs).

Multi-genre speaker verification. This task requires robustness over genre, platform, and device. The 'baseline' uses a conventional input normalization method in SV, global mean subtraction. We measure the performance by equal error rates (EER) (lower is better) on both the overall validation set and each genre validation set except 'advertisement,' 'play,' and 'recitation,' which consists of a very few positive pairs following the settings of Li et al. (2020) and Kang et al. (2020). We show the results in Table 3. The conventional normalization method produces poor generalization for novel genres, *i.e.*, 'movie' and 'singing.' RFN is also helpful for this metric learning task. RFN does not show dramatic improvements for unseen but still brings better genre robustness and the best overall EER of 13.4%.

5.3 Importance of Frequency-wise Normalization

Effectiveness of FN. Figure 3 left shows estimates of $I(\boldsymbol{s}^{(F)}, d)$ of BC-ResNet-1 on the ASC task. Here we compare 'Relaxed-IN' which use IN instead of IFN. We apply RFN or Relaxed-IN at the input and after every stage and get $\boldsymbol{s}^{(F)}$ from the activation of each stage (before RFN or Relaxed-IN).

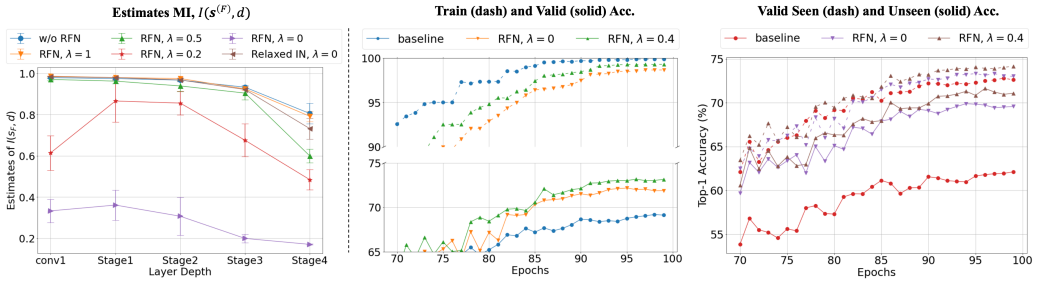

Figure 3: **Left: Estimates of mutual information**, $I(\boldsymbol{s}^{(F)}, d)$ where $\boldsymbol{s}^{(F)}$ and d are frequency-wise statistics of intermediate hidden feature and audio-device (domain) label, respectively. **Middle:** Train and validation accuracy curves. **Right:** Validation accuracies for seen and unseen domains. (Average over 5 seeds; error bar stands for std)

Then, we directly compare the effect of IFN and IN, which correspond to RFN and Relaxed-IN with $\lambda = 0$, respectively. RFN with $\lambda = 0$ dramatically reduces $I(\boldsymbol{s}^{(F)}, d)$ as expected, while the effect of Relaxed-IN is much smaller compared to RFN. We also confirm that relaxation can alleviate the information loss using bigger $\lambda \in \{0.2, 0.5, 1.0\}$.

Figure 3 middle and right show training curves for the larger network, BC-ResNet-8. In Figure 3 middle, the vanilla BC-ResNet-8 (baseline) and using Relaxed-IN show a large generalization gap between train and validation accuracies, and using RFN reduces the generalization gap with higher validation accuracy. Figure 3 right shows validation accuracies for seen and unseen devices. While the baseline and Relaxed-IN report poor validation accuracy for unseen devices, RFN boosts the performance for unseen. We also report the training curves of BC-ResNet-1 in Appendix A.5.

Table 4: **Compare frequency-wise approaches to their channel-wise counterparts** (averaged over 5 seeds).

ASC, top-1 accuracy (%) Method	seen	unseen	Overall
Baseline	72.3	62.0	68.9 ± 0.8
Global Norm	71.8	60.3	68.0 ± 0.7
BIN	70.2	60.1	66.8 ± 1.5
MixStyle	70.5	57.6	66.2 ± 0.7
Relaxed-IN	72.9	60.7	68.8 ± 0.6
Global FreqNorm	73.2	62.5	69.7 ± 0.6
BIFN	72.0	62.6	68.8 ± 1.0
Freq-MixStyle	73.2	67.7	71.3 ± 0.9
RFN (Ours)	**75.4**	**70.8**	**73.9 ± 0.7**
SV, EER (%) Method	seen	unseen	Overall
Baseline	11.8	28.2	14.5 ± 0.4
MixStyle	11.4	27.7	14.0 ± 0.3
Relaxed-IN	11.3	27.3	13.9 ± 0.3
Freq-MixStyle	**10.3**	**25.4**	**12.9 ± 0.7**
RFN (Ours)	10.6	26.9	13.4 ± 0.3

Frequency-wise vs. Channel-wise approaches. We further compare various channel-wise approaches to their frequency-wise counterparts for the ASC and SV tasks as in Table 4. 'Global Norm' normalizes inputs using channel-wise global statistics over training dataset, and its frequency-wise version is 'Global FreqNorm.' We use IFN instead of IN to get 'BIFN' from BIN, and 'Freq-MixStyle' mixes frequency-wise statistics rather than channel-wise statistics. We apply Freq-MixStyle at the input and after every stage in BC-ResNet-8 like RFN. We also use 'Relaxed-IN' as a counterpart of RFN. The results show consistent improvements for frequency-wise approaches compared to each of their channel-wise version. Thus, the results support the importance of frequency-wise approaches in audio, and RFN still outperforms most frequency-wise versions of the baselines except 'Freq-MixStyle' in SV, which shows a better result than our RFN with default $\lambda = 0.5$.

5.4 Ablation Studies

Complementariness of RFN. RFN is an explicit normalization and can be complementary to conceptually different DG approaches. Non-IN-based baselines, e.g., CrossGrad, CSD, and Mixup, improve the performance in KWS and SV, but they do not show clear improvements in ASC. Thus, we experiment with our RFN combined with non-IN-based baselines on KWS and SV. In KWS, CrossGrad and CSD show 0.5 to 1% improvements, and Mixup shows 0.4% improvements in SV compared to each of their vanilla training. In Table 5, RFN used in conjunction with each of them achieves additional improvements.

Table 5: **Top-1 test accuracy (%)** using RFN with other baselines (average over 5 seeds).

KWS Method	50	100	200
CrossGrad	73.7	81.1	87.2
CSD	73.4	80.9	87.5
RFN	76.6	82.3	87.8
RFN + CrossGrad	77.5	82.3	87.7
RFN + CSD	77.2	83.0	88.4
SV Method	seen	unseen	all
Mixup	11.1	28.3	14.1
RFN	10.6	26.9	13.4
RFN + Mixup	10.0	26.9	12.8

Table 6: **Position of RFN modules.** Compare Top-1 validation accuracy (%) (or EER (%) for SV) for the three tasks with number of multiplies. The numbers are average and standard deviation (average over 5 seeds).

ASC			SV			KWS		
RFN position	Top-1 Acc. (%)	#Mult	RFN position	EER (%)	#Mult	RFN position	EER (%)	#Mult
Baseline	68.9 ± 0.8	516.5M	Baseline	14.5 ± 0.4	451.8M	Baseline	84.4 ± 1.1	935.3M
Input	73.1 ± 0.7	516.7M	**Input**	$\mathbf{13.1 \pm 0.4}$	451.8M	**Input**	$\mathbf{86.3 \pm 1.0}$	935.3M
Input, Stage1	73.0 ± 0.6	519.9M	Input, Stage1	13.5 ± 0.2	451.9M	Input, ResBlock1,2	86.2 ± 0.6	936.7M
Input, Stage1, 2	72.7 ± 0.5	521.1M	Input, Stage1, 2	13.6 ± 0.4	452.0M	Input, ResBlock1,2,3,4	86.2 ± 0.6	938.1M
Input, Stage1, 2, 3	73.0 ± 0.8	521.5M	Input, Stage1, 2, 3	13.3 ± 0.2	452.0M	**Input, All ResBlocks**	$\mathbf{86.3 \pm 1.0}$	939.9 M
Input, Stage1, 2, 3, 4	$\mathbf{73.9 \pm 0.7}$	522.0M	Input, Stage1, 2, 3, 4	13.4 ± 0.3	452.1M	-		
Stage1, 2, 3, 4	69.5 ± 0.9	521.8M	Stage1, 2, 3, 4	14.3 ± 0.3	452.1M	All ResBlocks	85.2 ± 0.9	939.9M
substitute BN	68.5 ± 0.5	516.5M	substitute BN	14.6 ± 0.3	451.8M	substitute BN	84.2 ± 0.5	937.6M

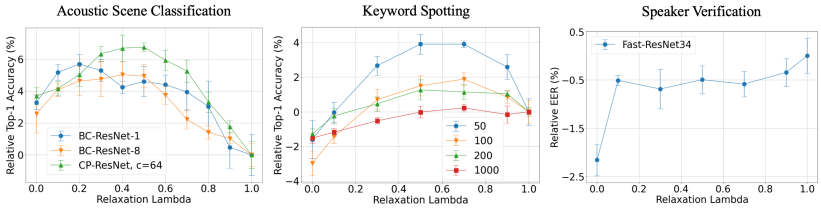

Figure 4: **Relative results** as λ changes compared to $\lambda = 1$ with various architectures (in ASC) or with varying number of train domains (in KWS) (Average over 5 seeds; error bar stands for std).

Position of RFN. So far, we naively apply RFN at the input and after every *stage* in ResNet-based architectures with naive relaxation of $\lambda = 0.5$. Table 6 shows experimental results and the number of multiplies of three tasks with the varying position of RFN on BC-ResNet-8, Fast-ResNet34, and ResNet15, respectively. Applying RFN to the input is the most effective compared to other positions in all tasks, and using RFN in deeper layers can bring additional gain in BC-ResNet-8. Also using RFN at other positions except input shows better results compared to the vanilla baselines. The result fits our observations in section 2 that $I(\boldsymbol{s}^{(F)}, d)$ is especially dominant at the input, and the literature of Pan et al. (2018) which reports that domain-wise feature divergence is higher in shallow layers. Here, we suggest using RFN at the input and after every stage as a good start. We also experiment with 'Substitute BN,' which switches all BN by RFN in the networks, and it results in poor performance compared to our suggestion.

Sensitivity to the degree of relaxation. In Figure 4, we plot how the varying λ affects the performance. For clarity, we draw the graphs of relative performance compared to $\lambda = 1$ for the experiments of Section 5.2. In both ASC and KWS, the optimal λ is between 0 and 1 for various architectures (ASC) and for varying number of training domains (KWS) which implies the necessity of relaxation in RFN. Especially in KWS, $\lambda = 0$ results in a performance drop. See Appendix A.5 for more results. On the other hand, SV shows a different tendency and gets the best EER of 11.7 $\pm$ 0.3% at $\lambda = 0$ (lower is better). One possible reason would be overfitting to domain information with larger λ due to the severe condition (high correlation), Cramér's V (Cramér, 2016) of 0.825, between class and domain labels of the CN-Celeb dataset. Compared to that, the other datasets used in ASC and KWS show 0.004 and 0.158, respectively. We believe that fixed λ is not optimal for every position and leave the automatic optimization of λ of RFN for future work. We report results using direct SGD or meta-learning for λ in Appendix A.5.

6 Conclusion

In this work, we address the characteristics of audio features in 2D-CNNs and show a guideline to get the audio domain invariant feature. Frequency-wise distribution is highly correlated to domain information, and we can eliminate instance-specific domain discrepancy by explicitly manipulating frequency-wise statistics rather than channel statistics. Based on the analysis, we introduce a novel domain generalization method, Relaxed instance Frequency-wise Normalization (RFN), which consists of IFN and additional normalization for relaxation, LN, in this work. The method shows its feasibility on three different tasks addressing multiple domains: acoustic scene classification, keyword spotting, and speaker verification with a clear margin compared to previous approaches.

REFERENCES

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

John Bronskill, Jonathan Gordon, James Requeima, Sebastian Nowozin, and Richard E. Turner. Tasknorm: Rethinking batch normalization for meta-learning. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1153–1164. PMLR, 2020.

Weicheng Cai, Jinkun Chen, and Ming Li. Exploring the encoding layer and loss function in end-to-end speaker and language recognition system. In *Odyssey*, pp. 74–81. ISCA, 2018.

Simyung Chang, Hyoungwoo Park, Janghoon Cho, Hyunsin Park, Sungrack Yun, and Kyuwoong Hwang. Subspectral normalization for neural audio data processing. In *ICASSP*, pp. 850–854. IEEE, 2021.

Seokeon Choi, Taekyung Kim, Minki Jeong, Hyoungseob Park, and Changick Kim. Meta batch-instance normalization for generalizable person re-identification. *CoRR*, abs/2011.14670, 2020.

Joon Son Chung, Jaesung Huh, Seongkyu Mun, Minjae Lee, Hee Soo Heo, Soyeon Choe, Chiheon Ham, Sunghwan Jung, Bong-Jin Lee, and Icksang Han. In defence of metric learning for speaker recognition. *arXiv preprint arXiv:2003.11982*, 2020.

Harald Cramér. *Mathematical Methods of Statistics (PMS-9), Volume 9*. Princeton university press, 2016.

Y. Fan, J. W. Kang, L. T. Li, K. C. Li, H. L. Chen, S. T. Cheng, P. Y. Zhang, Z. Y. Zhou, Y. Q. Cai, and D. Wang. Cn-celeb: A challenging chinese speaker recognition dataset. In *ICASSP*, pp. 7604–7608. IEEE, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778. IEEE Computer Society, 2016.

Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen. Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020. URL `https://arxiv.org/abs/2005.14623`. Submitted.

Hee Soo Heo, Bong-Jin Lee, Jaesung Huh, and Joon Son Chung. Clova baseline system for the voxceleb speaker recognition challenge 2020. *arXiv preprint arXiv:2009.14153*, 2020.

Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss, and Kevin W. Wilson. CNN architectures for large-scale audio classification. In *ICASSP*, pp. 131–135. IEEE, 2017.

Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, pp. 1510–1519. IEEE Computer Society, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015.

Jiawen Kang, Ruiqi Liu, Lantian Li, Yunqi Cai, Dong Wang, and Thomas Fang Zheng. Domain-invariant speaker vector projection by model-agnostic meta-learning. *arXiv preprint arXiv:2005.11900*, 2020.

Byeonggeun Kim, Simyung Chang, Jinkyu Lee, and Dooyong Sung. Broadcasted Residual Learning for Efficient Keyword Spotting. In *Proc. Interspeech 2021*, pp. 4538–4542, 2021a. doi: 10.21437/Interspeech.2021-383.

Byeonggeun Kim, Seunghan Yang, Jangho Kim, and Simyung Chang. QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design. Technical report, DCASE2021 Challenge, June 2021b.

Khaled Koutini, Hamid Eghbal-zadeh, and Gerhard Widmer. Receptive-field-regularized CNN variants for acoustic scene classification. *CoRR*, abs/1909.02859, 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pp. 1106–1114, 2012.

Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, 1989.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, pp. 5543–5551. IEEE Computer Society, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, pp. 3490–3497. AAAI Press, 2018.

Lantian Li, Ruiqi Liu, Jiawen Kang, Yue Fan, Hao Cui, Yunqi Cai, Ravichander Vipperla, Thomas Fang Zheng, and Dong Wang. Cn-celeb: multi-genre speaker recognition. *CoRR*, abs/2012.12468, 2020.

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR (Poster)*. OpenReview.net, 2017.

Irene Martín-Morató, Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen. Low-complexity acoustic scene classification for multi-device audio: analysis of dcase 2021 challenge systems, 2021.

Mark D. McDonnell and Wei Gao. Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths. In *ICASSP*, pp. 141–145. IEEE, 2020.

Hyeonseob Nam and Hyo-Eun Kim. Batch-instance normalization for adaptively style-invariant neural networks. In *NeurIPS*, pp. 2563–2572, 2018.

Kamalesh Palanisamy, Dipika Singhania, and Angela Yao. Rethinking CNN models for audio classification. *CoRR*, abs/2007.11154, 2020. URL `https://arxiv.org/abs/2007.11154`.

Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV*, volume 11208 of *Lecture Notes in Computer Science*, pp. 484–500. Springer, 2018. doi: 10.1007/978-3-030-01225-0_29. URL `https://doi.org/10.1007/978-3-030-01225-0_29`.

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 *INTERSPEECH*, pp. 2613–2617. ISCA, 2019.

Hyunsin Park and Chang D. Yoo. Cnn-based learnable gammatone filterbank and equal-loudness normalization for environmental sound classification. *IEEE Signal Process. Lett.*, 27:411–415, 2020.

Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning*, pp. 5102–5112. PMLR, 2019.

Sai Samarth R. Phaye, Emmanouil Benetos, and Ye Wang. Subspectralnet - using sub-spectrogram based convolutional neural networks for acoustic scene classification. In *ICASSP*, pp. 825–829. IEEE, 2019.

Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient domain generalization via common-specific low-rank decomposition. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7728–7738. PMLR, 2020.

N. Vishnu Prasad and Srinivasan Umesh. Improved cepstral mean and variance normalization using bayesian framework. In *ASRU*, pp. 156–161. IEEE, 2013.

Tara N. Sainath and Carolina Parada. Convolutional neural networks for small-footprint keyword spotting. In *INTERSPEECH*, pp. 1478–1482. ISCA, 2015.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *ICLR (Poster)*. OpenReview.net, 2018.

Themos Stafylakis, Johan Rohdin, and Lukás Burget. Speaker embeddings by modeling channel-wise correlations. *CoRR*, abs/2104.02571, 2021.

Raphael Tang and Jimmy Lin. Deep residual learning for small-footprint keyword spotting. In *ICASSP*, pp. 5484–5488. IEEE, 2018.

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, pp. 4105–4113. IEEE Computer Society, 2017.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *NIPS*, pp. 831–838. Morgan Kaufmann, 1991.

Olli Viikki and Kari Laurila. Cepstral domain segmental feature vector normalization for noise robust speech recognition. *Speech Commun.*, 25(1-3):133–147, 1998.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. *arXiv preprint arXiv:1805.12018*, 2018.

Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018a.

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5265–5274, 2018b.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. In *IJCAI*, pp. 4627–4635. ijcai.org, 2021a.

Yulin Wang, Zanlin Ni, Shiji Song, Le Yang, and Gao Huang. Revisiting locally supervised learning: an alternative to end-to-end training. In *ICLR*. OpenReview.net, 2021b.

Yuxuan Wang, Pascal Getreuer, Thad Hughes, Richard F. Lyon, and Rif A. Saurous. Trainable frontend for robust and far-field keyword spotting. In *ICASSP*, pp. 5670–5674. IEEE, 2017.

Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018.

Yuxin Wu and Kaiming He. Group normalization. In *ECCV (13)*, volume 11217 of *Lecture Notes in Computer Science*, pp. 3–19. Springer, 2018.

Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel, and Marc Niethammer. Robust and generalizable visual representation learning via random convolutions. In *International Conference on Learning Representations*, 2021.

Gary Yeung, Ruchao Fan, and Abeer Alwan. Fundamental frequency feature normalization and data augmentation for child speech recognition. In *ICASSP*, pp. 6993–6997. IEEE, 2021.

Neil Zeghidour, Nicolas Usunier, Gabriel Synnaeve, Ronan Collobert, and Emmanuel Dupoux. End-to-end speech recognition from the raw waveform. In *INTERSPEECH*, pp. 781–785. ISCA, 2018.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR (Poster)*. OpenReview.net, 2018.

Yuyang Zhao, Zhun Zhong, Fengxiang Yang, Zhiming Luo, Yaojin Lin, Shaozi Li, and Nicu Sebe. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *CVPR*, pp. 6277–6286. Computer Vision Foundation / IEEE, 2021.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European Conference on Computer Vision*, pp. 561–578. Springer, 2020.

Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*. OpenReview.net, 2021.

A APPENDIX

A.1 PYTORCH-LIKE PSEUDO-CODE FOR RFN

Algorithm 2 Pytorch-like pseudo-code for RFN

procedure RFN(x, λ, ϵ):
    # x: input feature with shape [N, C, F, T].
    # $\lambda \in [0, 1]$: relaxation.
    # ϵ: small constant, e.g., 1e-5

    # Instance Frequency-wise Norm stats.
    mean_ifn = x.mean(dim=(1, 3), keepdim=True)
    var_ifn = x.var(dim=(1, 3), keepdim=True, unbiased=False) + ϵ
    std_ifn = var_ifn.sqrt()

    # Layer Norm stats.
    mean_ln = x.mean(dim=(1, 2, 3), keepdim=True)
    var_ln = x.var(dim=(1, 2, 3), keepdim=True, unbiased=False) + ϵ
    std_ln = var_ifn.sqrt()

    # RFN.
    x = λ * (x - mean_ln) / std_ln + (1 - λ) * (x - mean_ifn) / std_ifn
    return x

A.2 STATISTICS OF RFN OUTPUT

Using the notations in Section 3, it is straight that the output of equation 2 has the mean of $\hat{\mu}_i^{(F)} = \frac{\lambda}{\sigma_i}(\mu_i^{(F)} - \mu_i)$. We briefly explain how to get $\hat{\sigma}_i^{(F)} = \lambda \cdot \frac{\sigma_i^{(F)}}{\sigma_i} + (1 - \lambda)$.

The outputs of LN and IFN result to mean and std of $(\mu_i^{(F)} - \mu_i)/\sigma_i$ and $\sigma_i^{(F)}/\sigma_i$ and 0 and 1, respectively. Here $\mu_i^{(F)}$ and $\sigma_i^{(F)}$ are calculated over a set $S_i = \{k|k_N = i_N, k_F = i_F\}$ while μ_i and σ_i are from $S_i' = \{k|k_N = i_N\}$. Let us consider the elements $x_i \in S_i$. For $\mathbf{X} = \{x_i\}$ and its RFN outputs $\mathbf{Y}$, the variance of $\mathbf{Y}$, the weighted sum of LN and IFN, is

$$\mathrm{Var}(\mathbf{Y}) = \lambda^2(\frac{\sigma_i^{(F)}}{\sigma_i})^2 + (1 - \lambda)^2 + 2\lambda(1 - \lambda) \cdot \mathrm{Cov}(\mathit{LN}(\mathbf{X}), \mathit{IFN}(\mathbf{X})). \tag{3}$$

The covariance in the last term is represented by expectations over $x_i \in S_i$. Here, μ_i, σ_i, $\mu_i^{(F)}$, and $\sigma_i^{(F)}$ are constant over the set S_i.

$$\begin{aligned}
\mathrm{Cov}(\mathit{LN}(\mathbf{X}), \mathit{IFN}(\mathbf{X})) &= \mathbb{E}_{S_i}[(\frac{x_i - \mu_i}{\sigma_i} - \frac{\mu_i^{(F)} - \mu_i}{\sigma_i})(\frac{x_i - \mu_i^{(F)}}{\sigma_i^{(F)}} - 0)] \\
&= \frac{1}{\sigma_i \cdot \sigma_i^{(F)}} \cdot \mathbb{E}_{S_i}[(x_i - \mu_i)(x_i - \mu_i^{(F)})] - \frac{\mu_i^{(F)} - \mu_i}{\sigma_i \cdot \sigma_i^{(F)}} \cdot \mathbb{E}_{S_i}[x_i - \mu_i^{(F)}] \\
&= \frac{1}{\sigma_i \cdot \sigma_i^{(F)}}(\mathbb{E}_{S_i}[x_i^2] - (\mu_i + \mu_i^{(F)}) \cdot \mathbb{E}_{S_i}[x_i] + \mu_i \cdot \mu_i^{(F)}) \\
&= \frac{1}{\sigma_i \cdot \sigma_i^{(F)}}(\mathbb{E}_{S_i}[x_i^2] - (\mu_i^{(F)})^2) = \frac{\sigma_i^{(F)}}{\sigma_i},
\end{aligned} \tag{4}$$

using $\mathbb{E}_{S_i}[x_i] = \mu_i^{(F)}$. Apply the result to equation 3, and $\mathrm{Var}(\mathbf{Y}) = (\lambda\frac{\sigma_i^{(F)}}{\sigma_i} + (1 - \lambda))^2$.

A.3 DESIGN CHOICE OF RFN

We use LN with IFN to alleviate the effect of IFN. There are other choices instead of LN, and we compare an identity shortcut, BN, and GN in Figure A1. BN and GN do not use affine transforma-

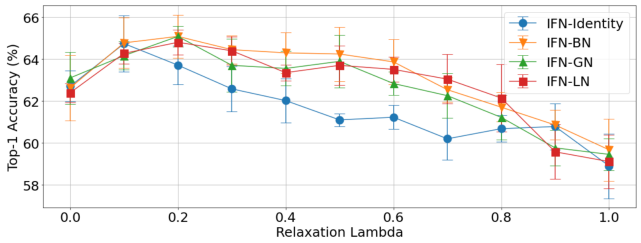

Figure A1: **Top-1 Validation Error (%)** on TAU Urban AcousticScenes 2020, development dataset with various tactics to relax IFN (Average over 5 seeds).

tion, and an identity shortcut stands for $\mathit{RFN}(\boldsymbol{x}) = \lambda \cdot \boldsymbol{x} + (1 - \lambda) \cdot \mathit{IFN}(\boldsymbol{x})$. We experiment with BC-ResNet-1 on acoustic scene classification task as in Section 5. The result shows that RFN with an identity shortcut also works. However, if we use the identity $\boldsymbol{x}$, the effect of λ depends on the distribution of $\boldsymbol{x}$. In other words, the role of λ can change by the distribution of $\boldsymbol{x}$. Thus, rather than using identity, we prefer using normalized features. Using BN or GN instead of LN shows similar results. However, BN has its weakness on small mini-batch size Wu & He (2018), and BN and GN also have additional parameters, running mean/var. Thus, in this work, we use LN with IFN as RFN, but still, we show that normalization like BN or GN also works well, and the choice can be changed depending on a task and a model architecture.

A.4 Details of Experimental Setup

Auxiliary Classifier. The auxiliary classifier in Section 2 has two fully connected (FC) linear layers with batch normalization (BN) and ReLU nonlinearity which is in the form of FC-BN-ReLU-FC. The number of hidden features is 32 in common, and BN uses affine transformation. After training a base model, we freeze it and train randomly initialized auxiliary classifiers, which use $\boldsymbol{s}$, statistics of an intermediate hidden feature, as inputs to predict the target labels, i.e., domain or class labels. Kim et al. (2021b) recently introduced a modified version of BroadCasting-Residual Networks (BC-ResNets) (Kim et al., 2021a), and won the task in the DCASE2021 challenge (Martín-Morató et al., 2021). We use the modified BC-ResNet-1 as a base ASC model.

Acoustic Scene Classification. In the training data, device 'A' has 10,215 samples while others (B, C, S1-S3) have 750 samples each. In the validation data, all devices from 'A' to 'S6' have 330 segments for each. Each recording is 10-sec-long, and the sampling rate is 48kHz. We do downsampling by 16kHz and use input features of 256-dimensional log-Mel spectrograms with a window length of 130ms and a frameshift of 30ms. During training, we augment data as follows: (1) In the temporal dimension, we randomly roll each input feature in the range of -1.5 to 1.5 sec, and the out-of-range part is concatenated to the opposite side; (2) We use Specaugment (Park et al., 2019) with two frequency masks and two temporal masks with mask parameters of 40 and 80, respectively, except time warping. We apply Specaugment only for the large model, BC-ResNet-8, and use it after the first RFN at input features. We train each model for 100 epochs using stochastic gradient descent (SGD) optimizer with momentum set to 0.9 and weight decay to 0.001. We use the mini-batch size of 100 and 64 for BC-ResNet-1 and 8, respectively. The learning rate linearly increases from 0 to 0.1 and 0 to 0.06 over the first five epochs as a warmup (Goyal et al., 2017) for BC-ResNet-1 and 8, respectively. Then it decays to zero with cosine annealing (Loshchilov & Hutter, 2017) for the rest of the training. We report the validation performance of each trial after the last epoch.

Keyword Spotting. Each utterance in Google speech commands dataset ver1 is 1-sec-long, and the sampling rate is 16kHz. We follow the official 12-class classification setting of Warden (2018). The dataset has total thirty words and we use ten words, "Yes", "No", "Up", "Down", "Left", "Right", "On", "Off", "Stop", and "Go" for ten class labels. There are two additional classes "silence" and "unknown word" consists of the rest twenty words. We borrow the settings introduced by Piratla et al. (2020)[1] for the experiments. We use 40-dimensional MFCCs with 30ms window and 10ms frame shift. We augment the input during the training using time-shift of T milliseconds where $T \sim \mathit{Uniform}[-100, 100]$ and adding background noise to each sample with a probability of 0.8

[1] https://github.com/vihari/CSD

Table A1: **Details of BC-ResNets** with input shapes where B, c and T stand for the mini-batch size, the number of base channel, and the total time steps, respectively.

Input: log-Mel spectrogram of B×1×256×T
RFN
Input:B×1×256×T, **conv1:**Conv5x5 with stride2 - BN - ReLU
Input:B×2c×128×T/2, **stage1:**BC-ResBlocks × 2
RFN
Maxpool 2×2
Input:B×c×64×T/4, **stage2:**BC-ResBlocks × 2
RFN
Maxpool 2×2
Input: B×1.5c×32×T/8, **stage3:**BC-ResBlocks × 2
RFN
Input: B×2c×32×T/8, **stage4:**BC-ResBlocks × 3
RFN
Input: B×2.5c×32×T/8, Conv1x1 - BN
Input: B×10×32×T/8, Global AvgPool
Output shape: B×10×1×1

following common settings (Tang & Lin, 2018; Piratla et al., 2020; Kim et al., 2021a). For the experiment of ResNet15, we follow the settings of Tang & Lin (2018). We use 40-dimensional MFCCs with 30ms window and 10ms frame shift and use same data augmentation as the experiment of cnn-trad-fpool3. We train the model for 9,000 iterations with mini-batch size of 64 using SGD optimizer with momentum set to 0.9 and weight decay to 1e-5. We use step decaying of learning rate whose initial value is 0.1 and we multiply by 0.1 to the learning rate at every 3,000 iterations.

Speaker Verification. Utterances in CN-Celeb are recorded in various lengths, and their sampling rate is 16kHz. During training, we use a fixed-length 2-sec-long temporal segment extracted randomly from each utterance. We convert the fixed segment into 40-dimensional log-Mel spectrograms for inputs of Fast ResNet-34. We train each model for 100 epochs using stochastic gradient descent (SGD) optimizer with momentum to 0.9 and weight decay to 0.001. We use a mini-batch size of 200, and the learning rate is decayed 0.1 to 0 with cosine annealing. We report the validation EER of each trial after the last epoch. We evaluate the trained models on the CN-Celeb validation set (CN-Celeb (E)). We sample ten 3-second temporal crops at regular intervals from each enroll and validation segment and compute cosine similarities between all possible combinations from every pair of segments. We use the average score of 100 similarities. This protocol is in line with Chung et al. (2020). We evaluate validation performance on each domain and overall domains in Table 3, and EER on seen/unseen domains are also reported in Table 4. We follow the official evaluation setting in CN-Celeb (Fan et al., 2020; Li et al., 2020). The overall validation set consists of million pairs of enrollment and validation utterances, which are various scenarios containing both the same genre and cross-genre. We split the overall validation set into each domain, and we report the performance of six seen domains and two unseen domains in Table 3. Since the validation sets of the rest domains are composed of very few positive pairs, 20, 50, and 105 for 'advertisement,' 'play,' and 'recitation,' respectively, we skip reporting EER for them following the settings of the previous works (Li et al., 2020; Kang et al., 2020). Similar to the setting for each domain, we get validation pairs of seen and unseen domains by splitting the overall validation set to measure the performance of seen and unseen genres.

Baseline Details. PCEN uses hyperparameters of $s = 0.025$, $\alpha = 0.98$, $\delta = 2$, and $r = 0.5$ following the settings of Wang et al. (2017). We use PCEN instead of log-Mel spectrogram. We use $\alpha = 0.2$ to get $\lambda \sim \text{Beta}(\alpha, \alpha)$ in Mixup (Zhang et al., 2018). Following the settings of Zhou et al. (2021), we apply MixStyle after stage 1 and 2 with $\alpha = 0.1$ and use the 'Random' shuffle and a probability of 0.5 to decide using MixStyle or not. For CSD, we use $k = 1$ and 2, the rank of the domain-specific component, when number of training speakers are $\{50, 100\}$ and $\{200, 1000\}$, respectively, in KWS, and we use $k = 1$ for the ASC task. Also, we use $\lambda = 1$, and $\kappa = 1$ following the default settings in Piratla et al. (2020). We choose $\alpha = 0.5$ and $\epsilon = 0.5$ for CrossGrad following

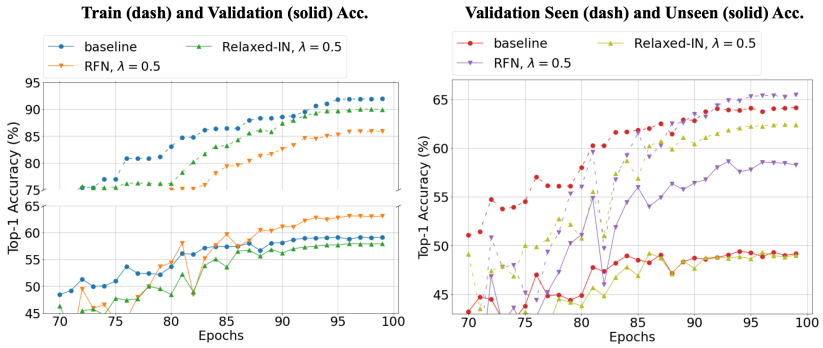

Figure A2: Experiment BC-ResNet-1. **Left:** Train and validation accuracy. **Right:** Validation accuracies of seen and unseen devices. (Average over 5 seeds).

the suggestion of Shankar et al. (2018). We use the official implementations[2] of Mixup, Mixtyle, BIN, and CSD.

Table A2: **Details of Fast-ResNet34** with input shapes where B and T stand for the mini-batch size and the total time steps, respectively.

Input: log-Mel spectrogram of B×1×40×T
RFN
Input:B×1×40×T, **conv1:**Conv7x7 with stride(2,1) - BN - ReLU
Input:B×16×20×T, **stage1:**ResBlocks × 3
RFN
Input:B×16×20×T, **stage2:**ResBlocks × 4
RFN
Input: B×32×10×T/2, **stage3:**ResBlocks × 6
RFN
Input: B×64×5×T/4, **stage4:**ResBlocks × 3
RFN
Input: B×128×5×T/4, Freq-wise Avgpool
Input: B×128×1×T/4, Self-Attentive Pooling (Cai et al., 2018)
Input: B×128×1×1, Fully-connected layer
Output shape: Bx512

Model architectures. Tables A1 and A2 show the details of architectures with RFN. In Table A1, c is 10 and 80 for BC-ResNet-1 and 8, respectively. The details of the BC-ResBlock are described in the literature of Kim et al. (2021a), and the ResBlock used in Fast-ResNet34 are described by Chung et al. (2020), and Heo et al. (2020). The details of cnn-trad-fpool3 (Sainath & Parada, 2015) is straight because RFN is only applied at the input.

A.5 More Experimental Results

Acoustic Scene Classification. In Table A3, we experiment with less seen domains. We do not use real device 'C' or 'B and C' during training. The result shows that RFN is helpful for better generalization of unseen real devices 'B' and 'C'. Table A4 shows more results on small model, BC-ResNets-1 and another ASC backbone, CP-ResNet (Koutini et al., 2019). We use the modified version of BC-ResNet-1 as indicated by Kim et al. (2021b).

Figure A2 compares training curves of vanilla BC-ResNet-1 to RFN on the ASC task.

[2]Mixup: `https://github.com/facebookresearch/mixup-cifar10`,
MixStyle: `https://github.com/KaiyangZhou/mixstyle-release`,
BIN: `https://github.com/hyeonseobnam/Batch-Instance-Normalization`,
CSD: `https://github.com/vihari/CSD`

Table A3: **Acoustic Scene Classification using Less Scene Domains.** Top-1 validation accuracy (%) on TAU Urban AcousticScenes 2020 Mobile, development dataset. We show mean and standard deviation of scores. (averaged over 5 seeds, *: average over 10 seeds)

		seen					unseen					
Method	#Param	A	B	S1	S2	S3	**C**	S4	S5	S6	Overall	Δ
BC-ResNet-8	315k	80.5	**71.4**	69.3	69.7	71.2	56.6	63.3	63.7	56.5	66.9 ± 1.1	+ 0.0
+ PCEN	315k	75.4	67.4	69.8	67.6	74.1	59.8	67.7	66.8	65.2	68.2 ± 0.5	+ 1.3
+ Mixup	315k	79.2	67.4	69.8	64.4	69.9	56.3	61.3	60.3	53.3	64.7 ± 1.1	- 2.2
+ MixStyle	315k	79.9	65.7	67.9	65.7	71.0	56.2	67.0	67.0	61.8	66.9 ± 0.8	+ 0.0
+ BIN	317k	76.8	68.6	66.1	64.5	70.1	53.2	62.1	62.8	56.1	64.5 ± 0.9	- 2.4
+ Freq-MixStyle	315k	80.9	70.1	71.3	70.9	75.6	61.8	69.0	69.6	62.0	70.1 ± 0.5	+ 3.2
+ RFN (Ours)	315k	**81.3**	70.9	**75.7**	68.4	**75.7**	**64.0**	68.2	**72.5**	**67.5**	**71.6 ± 0.4**	**+ 4.7**
$\lambda = 0$ (IFN)	315k	77.4	70.4	72.7	**71.9**	74.7	62.0	**72.0**	71.8	66.0	71.0 ± 0.6	+ 4.1
$\lambda = 1$ (LN)	315k	80.7	**71.4**	72.8	68.0	71.7	53.3	59.4	62.7	55.8	66.2 ± 0.7	-0.7

		seen				unseen						
Method	#Param	A	S1	S2	S3	**B**	**C**	S4	S5	S6	Overall	Δ
BC-ResNet-8	315k	**81.2**	70.4	67.1	70.8	41.4	54.2	62.7	64.4	54.8	63.0 ± 0.7	+ 0.0
+ RFN (Ours)	315k	80.4	**74.4**	**69.6**	**74.5**	**49.2**	**62.0**	**70.1**	**72.7**	**67.2**	**68.9 ± 0.7**	**+ 5.9**

Table A4: **More Results in Acoustic Scene Classification.** Top-1 validation accuracy (%) on TAU Urban AcousticScenes 2020 Mobile, development dataset. We show mean and standard deviation of scores. (averaged over 5 seeds)

Method	#Param	seen						unseen			Overall	Δ
		A	B	C	S1	S2	S3	S4	S5	S6		
BC-ResNet-1	8.1k	73.3	61.3	**64.9**	61.0	58.3	66.7	51.8	51.3	48.5	59.7 ± 1.3	+ 0.0
+ Global FreqNorm	8.1k	72.2	59.5	62.7	59.3	56.8	63.9	49.3	51.1	45.2	57.8 ± 1.2	- 1.9
+ PCEN	8.1k	68.4	55.5	58.9	60.1	57.6	63.9	59.3	61.9	56.8	60.3 ± 1.1	+ 0.6
+ Mixup	8.1k	72.4	61.1	63.2	58.5	56.9	63.7	49.5	51.6	44.5	57.9 ± 1.6	- 1.8
+ MixStyle*	8.1k	71.6	54.8	58.8	53.1	53.5	57.6	43.0	46.0	39.0	53.0 ± 1.5	- 6.7
+ BIN*	8.3k	74.7	60.6	**64.9**	59.3	59.9	65.9	54.6	53.3	49.5	60.3 ± 2.0	+ 0.6
+ CSD*	8.4k	74.6	60.7	64.7	59.9	57.4	66.9	50.8	51.5	46.6	59.2 ± 1.2	- 0.5
+ Freq-MixStyle*	8.1k	71.7	59.5	59.1	58.3	57.5	63.9	58.9	60.1	52.0	60.1 ± 1.0	+ 0.4
+ BIFN*	8.3k	74.8	62.2	63.1	59.9	58.0	65.1	54.1	55.1	50.4	60.3 ± 1.3	+ 0.6
+ RFN (Ours)	8.1k	**75.2**	**63.7**	64.0	**62.8**	61.2	**68.0**	58.3	**63.0**	**57.2**	**63.7 ± 0.9**	**+ 4.0**
$\lambda = 0$ (IFN)	8.1k	67.6	61.5	57.7	63.6	**64.1**	65.3	**63.3**	61.6	56.8	62.4 ± 0.5	+ 2.7
$\lambda = 1$ (LN)	8.1k	73.8	60.2	61.7	61.0	59.4	65.2	51.3	53.5	45.9	59.1 ± 1.3	- 0.6
Relaxed-IN	8.1k	73.3	58.9	59.0	59.2	59.3	62.5	52.1	51.6	44.5	57.8 ± 0.8	- 1.9
CP-ResNet, $c = 64$	897k	78.1	**71.2**	**73.4**	68.3	65.9	68.7	64.8	64.8	58.5	68.2 ± 0.4	+ 0.0
+ RFN using BN	897k	**80.4**	72.8	73.6	**73.8**	71.5	**75.1**	**72.4**	**70.7**	**70.2**	**73.4 ± 0.3**	**+ 5.2**
$\lambda = 1.0$ (BN)	897k	77.3	**72.9**	**73.9**	68.6	66.1	68.4	65.3	63.9	57.8	68.2 ± 0.3	+ 0.0
+ RFN using LN	897k	79.3	70.9	70.8	71.8	**72.1**	74.1	69.9	68.6	66.0	71.5 ± 0.3	+ 3.3
$\lambda = 1.0$ (LN)	897k	73.7	68.8	67.5	63.8	65.8	68.1	59.6	61.8	53.6	64.7 ± 0.8	- 3.5

Figure A3 left shows sensitivity to relaxation λ on BC-ResNet-8 with varying number of domains, 2, 4, and 6. We use only 'A' and 'S1' for the number of domains of 2 and 'A,' 'B,' 'S1,' and 'S2' for the number of domains of 4. 6 is the default setting we've used in Section 5. Regardless of domain size, the optimal λ is between 0 and 1. Figure A3 right compares RFN to its channel-wise version, Relaxed-IN, which uses IN instead of IFN. When $\lambda = 0$, training of Relaxed-IN fails due to the direct use of IN at the last stage.

Keyword Spotting In Table A5, we compare our method to PCEN. For fair comparison, we use log-mel spectrogram instead of MFCCs for vanilla network and RFN.

Automatic update of relaxation, λ. Automatic update of λ is not straightforward due to unseen domains during training and imbalance between seen domains. Table A6 compares various tactics to decide λ. First, based on the observations in Figure 1, we gradually increase λ as layers go deep. We naively use $\lambda = [0.1, 0.3, 0.5, 0.7, 0.9]$ or $\lambda = [0.5, 0.6, 0.7, 0.8, 0.9]$ for input, after stage 1, stage 2,

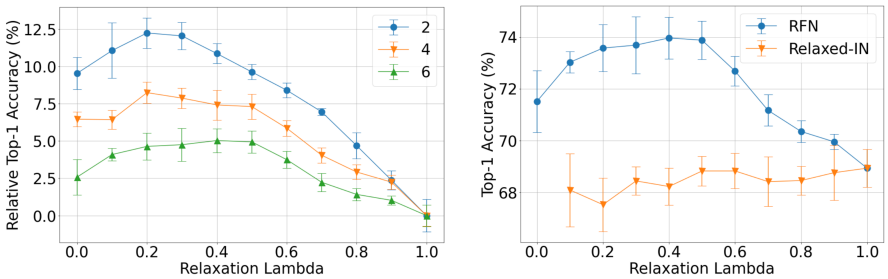

Figure A3: Top-1 test accuracy (%) of BC-ResNet-8 on TAU Urban Acoustic Scenes 2020 Mobile, development dataset. **Left** shows graphs with varying number of training domains 2, 4 and 6, and **Right** compares RFN with Relaxed-IN. (Average over 5 seeds; error bar stands for std)

Table A5: **Keyword Spotting. Compare PCEN to others using log-Mel spectrogram instead of MFCCs.** Top-1 test accuracy (%) with varying number of training speakers on Google speech command dataset ver1. (average and standard deviation; averaged over 5 seeds)

Method	50	100	200	1000
cnn-trad-fpool3	75.2 ± 0.3	79.8 ± 0.4	86.0 ± 0.3	92.0 ± 0.4
+ PCEN	81.6 ± 0.4	85.6 ± 0.5	89.5 ± 0.2	94.0 ± 0.1
+ RFN (Ours)	$\mathbf{82.6 \pm 0.3}$	$\mathbf{86.3 \pm 0.5}$	$\mathbf{90.8 \pm 0.2}$	$\mathbf{94.2 \pm 0.2}$

stage 3, and stage4, respectively. The linear interpolated λ results in 0.9% lower validation accuracy compared to that of fixed $\lambda = 0.5$. Second, we try a naive SGD update of λ, but it results in poor performance. Further, we tried the Meta-Learning Domain Generalization approach (MLDG) (Li et al., 2018). We use the same optimizer and learning rate schedule as the baseline and use $\alpha = \gamma$ and $\beta = 1$ for MLDG and use the first-order approximation of MAML (Finn et al., 2017). We trained the network parameters by conventional SGD training and updated λ by meta-test loss in the MLDG scenario. The approach got λ of [0.7, 0.5, 0.3, 0.5, 0.1] for input and after each stage, respectively, on average over five seeds and results in 72.2% accuracy, which is better than naive SGD but still worse than the fixed $\lambda = 0.5$. We leave the automatic update of λ as future work.

Table A6: **Decision rules for $\boldsymbol{\lambda}$.** Compare Top-1 validation accuracy (%) of BC-ResNet-ASC-8 on TAU Urban AcousticScenes 2020 Mobile, development dataset (average over 5 seeds).

Method	Top-1 Acc. (%)
Baseline (fixed $\lambda = 0.5$)	73.9 ± 0.7
+ Linear interpolation	73.0 ± 0.9
+ Naive SGD update	69.2 ± 0.5
+ Meta-learning domain generalization (MLDG)	72.2 ± 0.9

Analysis of Instance Statistics of 2D Audio Features. We also estimate MI while train the vanilla BC-ResNet-1 by frequency-wise pooling at last. We did the global avgpool at last in Table A1 along frequency-axis instead of channel, and it results to activation $\boldsymbol{x} \in \mathbb{R}^{N \times 1 \times F \times 1}$. We use frequency-wise pooled feature for classification. The Figure A4 shows the result that domain information in frequency-wise statistics still seems dominant compared to the other dimensions. Figure A5 shows more t-SNE visualizations which compare $\boldsymbol{s}^F$ and $\boldsymbol{s}^C$

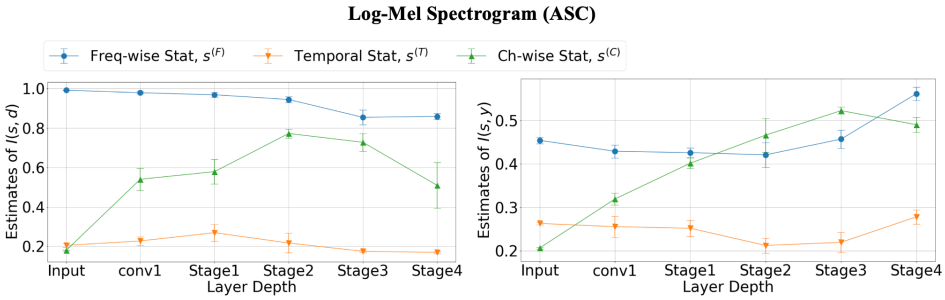

Figure A4: **Estimates of mutual information** between dimension-wise statistics of intermediate hidden feature and the domain label, $I(\boldsymbol{s}, d)$ (left), or the class label, $I(\boldsymbol{s}, y)$ (right). We use log-Mel spectrogram (Average over 5 seeds; error bar stands for standard deviation).

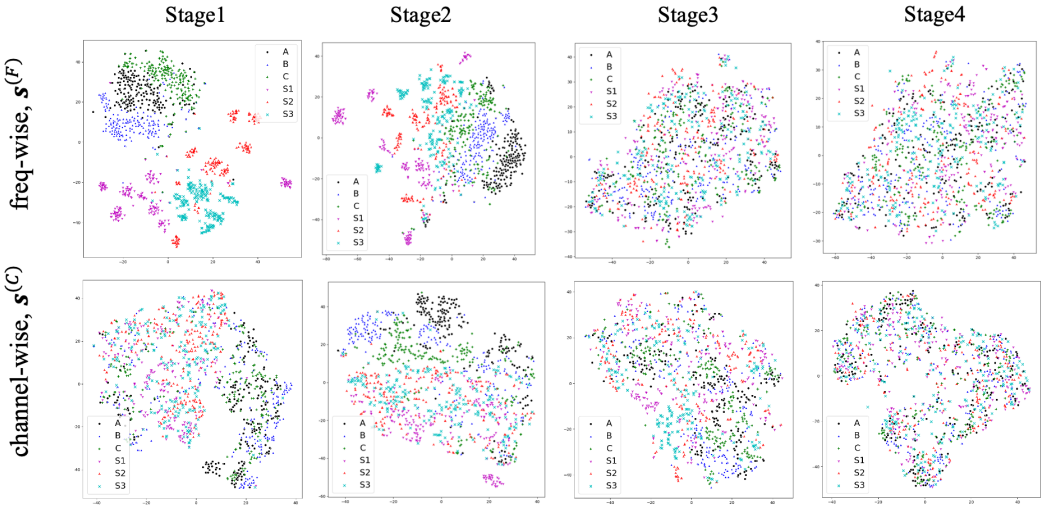

Figure A5: **2D t-SNE visualiations** using activations of Vanilla BC-ResNet-1.