Few-Shot Learning for Acoustic Event Detection

Saksham Gupta Sagnik Mukherjee D. K. Emmanuel Raju
170613, 170606, 170249
{gupsak, sagnikm, derkomal}@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

Abstract

In our work, we present a Few-Shot learning method for the classification of audio events. We take inspiration from the paper "Learning to Compare: Relation Network for Few-Shot Learning". In the report, we provide extensive results for our experiments using two datasets. We also discuss some future directions to work on based on our results and error analysis.

1 Introduction

In this paper, we address the classification of an audio signal containing a single event. Previous works exist which deal with this to a reasonable extent, including many state-of-the-art deep learning models. However, the data hunger of the systems often proves that they are no match for human intelligence. We try to build upon the existing concepts of meta-training and few-shot learning to address this issue. Few-shot learning is a common research problem in machine learning for quite some time now. For example, a person only needs to observe a few figurative elements of a group to learn its representation permanently; it takes a massive amount of data to train a deep learning model for the same job. Few-shot (or One/Zero) learning tries to tackle this issue. It is pretty common to see Few-Shot learning in the setup of **meta-learning**. At its core, we can characterise meta-learning by the need that these algorithms serve - given the experience on the set of previous tasks, we want to learn new tasks more quickly with a higher order of proficiency. Herein, we leverage the model's experience while classifying previous, relatively data-heavy classes to learn new classes predictions using a few labelled examples (shots).

2 Problem Description

Acoustic Event Detection (AED) is a multi-classification problem. The datasets consist of a weakly annotated audio clip - label pairs. In this paper, the datasets have some target classes that are significantly underrepresented (**class imbalance**) in the dataset. i.e. only a few examples of those are available in the training set. Therefore, creating an opportunity to apply for Few-Shot classification techniques and algorithms. In the subsequent section, we will look at our datasets; then compare the performance of some traditional deep learning models with a Few-Shot technique, **Relation Networks** [1].

3 Datasets and Pre-Processing

For all out experiments, we use two publically avialable datasets:**UrbanSound8K** [2]: 8,732 labelled excerpts of urban sounds from 10 classes. **DCASE** [3]: We use the dataset provided for task-1 in DCASE 2021 challenge. The dataset contains 23,040 (equally-distributed) recordings from 10 different acoustic scenes.

For both the datasets mentioned above, we create two custom datasets, **Class-imbalanced data** and **Few-Shot Data** for baseline and few-shot model, respectively. The description is as follows:

1. To create a massive class imbalance for our Class Imbalanced dataset, we do randomly sampling from both datasets. We randomly divide the ten labels into two sets, *sample* and *fewshot* of size 7 and 3, respectively (for UrbanSound8K, 'gun shot' and 'car horn' are always in *fewshot* as both have a limited number of examples). For *sample* classes, we randomly select 500/600 audio - label pairs and 20/5 for the *fewshot* classes. The test dataset consists of examples from *fewshot* classes only.

2. For the Few-Shot dataset, we randomly divide the ten labels into two sets, *sample* and *fewshot* of size 7 and 3, respectively. There are no examples from the *fewshot* classes during training. We divide the *samples* classes into mini-batches of 20/30 examples each. The test dataset consists of examples from *fewshot* classes only, again divided into min-batches of 20 examples.

The data is present in .wav format; to feed it to our models, we extract the Short-time Fourier transform (**STFT**) of the audio clips using the **librosa**[1] library. Next, each STFT is resized to 513×401, where 401 corresponds to an audio clip of 4 seconds. An STFT is either randomly truncated or zero-padded to obtain the required dimensions. We use one-hot encoding for labels.

4 Models

4.1 Baseline

We experiment with two baseline model whose architecture is as follows:

1. **Neural Network** (NN): The model has two layers - a hidden layer of 15 and an output layer of 10 (equal to the number of classes). For input, we extract 20 point MFCC features, taking an average along the time axis.

2. **Convolutional Neural Network** (CNN): Figure 1 shows the detailed architecture of the CNN based model [4]. A very naive approach to looking at the Few-Shot learning problem is to consider it a class imbalance problem. We tried a weighted loss function; it heavily penalizes the misclassification of the few-shot classes. We used the Cross-Entropy loss function with the Adam optimizer and saved the model with the best validation accuracy.

4.2 Few-Shot Model: Relation Network

We use the Few-Shot dataset to train the model. The training is divided into Episodes, where each episode mimics the few-shot training - the data is divided into **sample** and **query** examples. If the sample examples contain K labelled examples for each of C unique classes, the target few-shot problem is called C-way K-shot.

The model has two primary components - an **Embedding module** and a **Relation module**. The Embedding module consists of four "Convolutional-BatchNorm-ReLu-MaxPool" blocks. The

[1]`https://librosa.org/doc/latest/index.html`

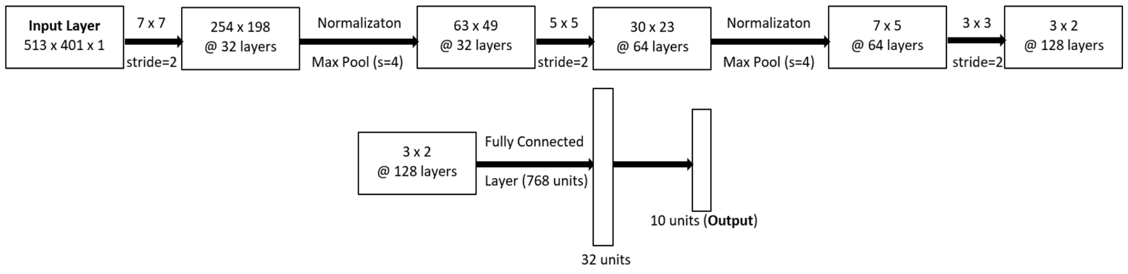
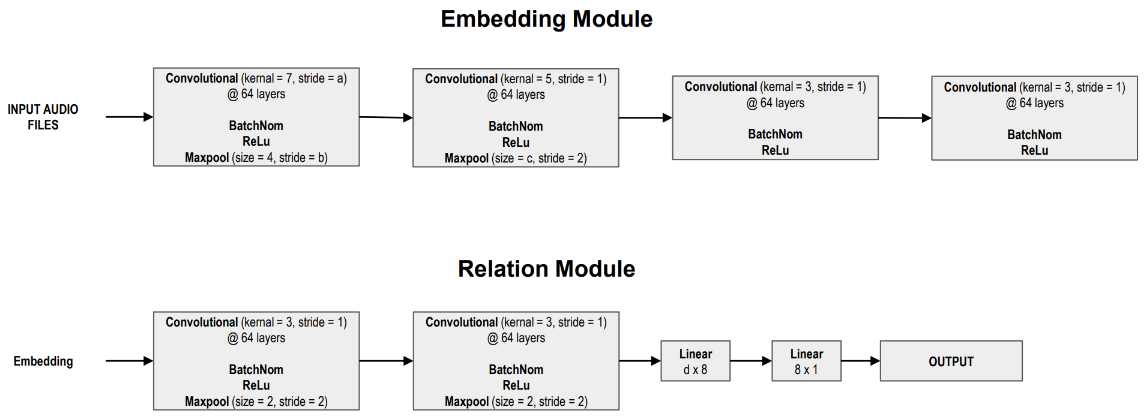


Figure 1: Model architecture for CNN based baseline model

relation network consists of 2 "Convolutional-BatchNorm-ReLu-MaxPool" blocks, followed by two linear layers [5]. Figure 2 shows the detailed architecture of the model. We train the model in an end-to-end fashion. The intuition for the model is that it creates a representation, called **class space**, of every class using the Embedding module (using the sample audios). Next, the model passes the query images through the Embedding module and compares them to every class space using the Relation module. The Relation module outputs the class to which the query audio matched the most.



Embedding Module
INPUT AUDIO FILES
Convolutional (kernal = 7, stride = a) @ 64 layers
BatchNom
ReLu
Maxpool (size = 4, stride = b)
Convolutional (kernal = 5, stride = 1) @ 64 layers
BatchNom
ReLu
Maxpool (size = c, stride = 2)
Convolutional (kernal = 3, stride = 1) @ 64 layers
BatchNom
ReLu
Convolutional (kernal = 3, stride = 1) @ 64 layers
BatchNom
ReLu
Relation Module
Embedding
Convolutional (kernal = 3, stride = 1) @ 64 layers
BatchNom
ReLu
Maxpool (size = 2, stride = 2)
Convolutional (kernal = 3, stride = 1) @ 64 layers
BatchNom
ReLu
Maxpool (size = 2, stride = 2)
Linear d x 8
Linear 8 x 1
OUTPUT

Figure 2: Model architecture for Relation Network
For 5-shot model: a = 2, b = 4, c = 2, d = 1536
For 1-shot model: a = 1, b = 2, c = 4, d = 40832

5 Experimental Details

We train our models on Tesla K-80 GPU at the Google Colaboratory. We train the baseline models using the Class-Imbalanced data. The training of NN baseline model is unstable for all hyperparameter settings; the accuracy is less than 5%. The CNN based model trains well only when we use weighted loss function. The results are summarized in Table 1.

For our Few-Shot model, we train two models: **3-way 5-shot** and **3-way 1-shot** using the Few-Shot data. The architecture of the Embedding module is smaller for the 5-shot model due to resource constraints. For the 1-shot and 5-shot model, the number of query audios is 10 and 15, respectively. We use Adam optimizer with a learning rate scheduler, it has a step size of 10 and gamma equal to 0.9). A standard Mean Squared loss function is used to calculate train loss. Table

2 summarises the results for various experiments of the Relation Network model.

Model Name	**Dataset**	**Accuracy**	**Weighted loss?**	**Minority class**
NN	UrbanSound8K	~2%	No	20
CNN	UrbanSound8K	~17%	No	20
CNN	**UrbanSound8K**	**63.58%**	**Yes**	**20**
CNN	UrbanSound8K	34.67%	Yes	5
CNN	DCASE	57.05%	Yes	20

Table 1: Result summary of Baseline models
Accuracy is on Test Data. Minority class means the number of example per minority class in Training Data.

Method	**Dataset**	**Train loss**	**Test Accuracy**
3-way 1-shot	UrbanSound8K	0.186	47.53%
3-way 5-shot	**UrbanSound8K**	**0.149**	**50.82%**
3-way 1-shot	DCASE	0.220	33.34%
3-way 5-shot	DCASE	0.156	46.00%

Table 2: Result summary for Relation Network

The **5-shot model is better than the CNN baseline** with 5 examples per minority class.

6 Error Analysis

- Google Colaboratory has **limited GPU resources**. It is not possible to train the model for more than a few hours. Also, for even 5-shot learning, we had to use a smaller model to fit GPUs' memory on Colab. Using better resources will undoubtedly improve performance.
- The performance of deep learning models is dataset dependent, i.e. an architecture that works for one may fail for another. In our case, even the domain is different (images to audio). Therefore, we carefully tune the embedding and relation modules and learning-rate scheduler. Yet, we are far from the 99% accuracy the model achieves image data.
- The idea of our model, Relation Network, is to learn the representation of classes, which typically requires many classes to be present in the dataset. However, that's not the case with our datasets (only ten classes), which may have led to an underwhelming performance.

7 Conclusion

The domain adaptation for any Deep earning model from images to audio is not a trivial task. However, using the basic intuition of the embedding module (to create a hidden representation of audio files) and relation network (learning from prototypes framework), one can design frameworks to work n audio clips. Another approach to work on is pre-training the embedding module (in the form of a classifier) and then training the relation network instead of end to end learning. This step by step training is comparatively more stable and converges better. In this paper, despite restricted resources and limited audio classes, we show that Few-Shot methods for AED can give better results than traditional data-hungry deep learning models.

References

[1] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," *CoRR*, vol. abs/1711.06025, 2017.

[2] J. Salamon, C. Jacoby, and J. P. Bello, "A Dataset and Taxonomy for Urban Sound Research," in *22nd ACM International Conference on Multimedia (ACM-MM'14)*, (Orlando, FL, USA), pp. 1041–1044, Nov. 2014.

[3] T. Heittola, A. Mesaros, and T. 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.

[4] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. Software available from tensorflow.org.

[5] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.