Semi Supervised Audio Event Detection using Tri-training

Aashi Pathak Neha Kumari Sumit Singhal
Team name: The vigilantes
Indian Institute of Technology
Kanpur, India

May 17, 2021

Abstract

In our study, we have developed methods to classify large amount of unlabeled data using small set of labeled data using semi supervised learning approaches like self-training, tri training and tri training with disagreement. The data used for this purpose is UrbanSound8k where we used only 30% of data in training and predicted 60% of unlabeled data.

1 Introduction and Related Work

Acoustic Event detection deals with processing acoustic signals and aims to classify them into different sound events. The supervised approach for classification requires a large amount of labeled data to learn relationships between independent and target variables. Often, labeling the data is time expensive and laborious thus it is not possible to have large labeled data all time. In many cases of AED, there are too few samples for rare events, such that classification of such events becomes problematic.

A possible solution is to combine the large unlabeled data with the help of small amount of labeled data. This area of study is known as semi-supervised learning where supervised methods are combined with unsupervised learning. In this approach, a small amount of labeled data is used to make predictions on the unlabeled data. Since these predictions are better than random guessing, thus they are treated as proxy labels. They are combined with the labeled data for subsequent iterations of the model. These proxy labels can be considered weak or noisy as they don't reflect the ground truth. The focus of our work is to leverage unlabeled audios to improve accuracy for acoustic event detection.

2 Data and Feature Extraction

We have used UrbanSound8k dataset to illustrate our purpose. The data has 10 classes: air conditioner, car horn, children playing, dog bark, and street music, gunshot, drilling, engine idling, siren, jackhammer. The dataset has about 8,732 audio segments of 3.5-sec average duration distributed into 10 stratified cross-validation folds. The idea revolves around utilizing the spectrogram image of the audio file and transforming them into data points. Firstly, we load each audio file and discretize it into a default sampling rate of 22.5kHz using the librosa load function. The function returns a time series of amplitudes of length 22.5kHz times the duration of each audio signal. Then we transformed the time series into a mel spectrogram where frequencies are converted to the mel scale. The input features are normalized before training.

3 Methods (self implemented)

To illustrate semi-supervised learning, we have taken 60% of the data and dropped its label to make it as unlabeled dataset. The rest 40% has been divided as 30% for training and 10% for testing. Although supervised techniques would give less accurate results with less data, we have taken supervised algorithms like CRNN, LSTM, MLP, and performed semi supervised methods like self and tri-training using these algorithms.

3.1 Self Training

We had implemented self-training as our baseline approach, it is an attempt to use unlabeled data to enhance the learning process. In this method, the model is trained in a supervised setting which is used to make predictions of unlabeled data. The confident predictions whose probability is greater than a threshold are added into training and the model is retrained. The process is repeated until no more predictions are confident. The pseudo code of the algorithm is shown in Figure 1.

Convolution Neural Networks (CNN)

To create the classifier we use 2 max pool layers of pool size =(2,2) and 2 convolution layers of kernel size = (3,3) and a number of filters 64 and 128 respectively. It is followed by a flatten layer, a dense layer of 1024 nodes, and an output layer of 10 nodes. For CNN, threshold of 0.9 is taken to add the pseudo label data into the training.We compared these proxy labels with their actual ones and found that approx 94% labels added are correct when threshold is 0.9. For 0.99 of threshold, the percentage correct is 97%. Therefore in LSTM, we added only those labels whose probability is greater than 0.99. Table 1 shows 5 iterations for self trained version of CNN.

Table 1: Iterations for Self Training of CNN

Itr	Training samples	Unlabeled	Train Acc	Val Acc	labels>0.9
1	2619	5240	83.196944	68.57506633	2035
2	4654	3205	91.89438224	82.39083886	614
3	5268	2591	91.13100171	83.93421769	146
4	5414	2445	91.5017128	83.63077044	123
5	5537	2322	92.79999733	84.17569399	123

LSTM

Long short term memory (LSTM) networks are well-suited for classifying, processing and making predictions based on time series data. The features extracted (in section 2) are trained on a sequential model which consisted of a sequence of layers. The first layer is LSTM which has 256 hidden nodes followed by flatten layer. This is followed by two dense layer of 128 nodes, in which the first one is proceeded with a dropout of 0.1. In the following Table 2 we have added only those proxy-labeled data to our training set whose probability of prediction of that class is greater than 99%. It can be seen that in nine iterations around 1000 unlabeled data has been annotated and are used for prediction purpose for nest iteration.

3.2 Tri-training

Tri training is a co-training style semi-supervised learning algorithm. This algorithm generates three classifiers from a labeled dataset. These classifiers are then refined using unlabeled examples in the

Table 2: Iterations for Self Training of LSTM

Itr	Training	Unlabeled	labels>0.99	Train Acc	Validation Acc
1	1833	5240	682	76.86852217	62.34096885
2	2310	4558	187	78.61471772	66.80120826
3	2441	4371	322	84.84227657	74.49856997
4	2667	4049	110	84.8518908	72.96587825
5	2744	3939	83	83.70991349	76.02040768
6	2802	3856	22	82.83368945	73.10574651
7	2817	3834	43	85.09052396	74.75165725
8	2847	3791	30	84.43976045	75.42997599
9	2868	3761	64	86.71548367	77.72357464

tri-training process. This method is better than self training as it provides better approximation of proxy labels for unlabeled data as data is added based on agreement of multiple models. The main requirement of tri-training is that the three models should be as diverse as possible. Hence different model architectures and bootstrapping of the sample data before feeding into training for each model is employed. In each round of tri-training,t. Data x is considered as a pseudo-label candidate for class c of one model if its probability output by other two models is sufficiently high. The pseudo-code for the algorithm is shown in Figure 2

CRNN

The Convolutional Recurrent Neural Networks is the combination of two of the most prominent neural networks,i.e. CNN(convolutional neural network) followed by the RNN(Recurrent neural networks). Here for RNN network we are taking LSTM model. To create the classifier we used 2 max pool layers of pool size =(2,2) and 2 convolution layers of kernel size = (3,3) and a number of filters 64 and 128 respectively. It is followed by 1 LSTM layer, a Dense layer of 32 nodes and a final output layer (dense) of 10 nodes.

MLP

Deep Neural Multilayer Perceptron (MLP) is a type of artificial neural network (ANN). Simplest MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. In the code, we have modeled two hidden dense layers with different neurons (512,256) in each layer. Considering the input and output layer, we have a total of 4 dense layers in the model. In case any optimiser is not mentioned then "Adam" is the default optimiser.For classification scenarios, the cross-entropy is the loss function for the classification. Here we have used learning rate of 0.001.

LSTM

Same model as in section 3.1.2 is used here.

We had illustrated two variants of tri-training algorithm as follows.

3.2.1 Tri-training With Majority Votes

In tri-training with majority votes[7], a data is considered as pseudo label if the probability for that class from any two models is greater than threshold (0.9 in case). This pseudo labeled data is added into the labeled set and removed from the unlabeled set for each model. This data augmentation

process has been illustrated for 6 iterations in Table 3. The confusion matrix is shown in Figure 4 (Figure 4)

Table 3: Iterations for Tri-training with majority votes

itr.	labled	unlabled	labels added	% acc.	CRNN val acc.	LSTM val acc.	MLP val acc.	test acc.
0	2619	5240	2526	92.201	74.504	86.260	81.221	74.341
1	5145	2714	373	83.378	79.021	86.014	81.585	75.143
2	5518	2341	223	85.650	82.319	86.159	82.174	76.403
3	5741	2118	85	82.353	79.109	87.396	84.192	74.685
4	5826	2033	160	78.125	82.773	86.822	85.518	75.830
5	5986	1873	90	67.778	82.899	85.705	82.565	76.174

3.2.2 Tri-training With Disagreement

In this approach, the model is strengthened only at its weak points. Here, data x is considered a pseudo labeled for a class c when the other two models have prediction probability greater than threshold (0.6 in this case) for that class while the main model has probability lower than a threshold (0.3 in this case). The method is more data efficient as it adds only those data to the training on which the model disagrees and the other two model agrees. This particular approach is known as tri-training with disagreement. At each iteration, the models are updated using the pseudo labeled data.

We have taken a hypothesis that at each iteration the mis-classification error rate of each model must be less than the previous iteration. Let z be the number of examples on which two models show same label and x be the correct number of those labels. Then, error would be z-x/ z. The error is taken to illustrate the diversity of models after each iteration, which is a basic requirement of tri training. In table 4, it can be seen that in successive iterations the error has been decreasing which proves the correctness of our algorithm. The '% correct' column in the following table shows the percentage of correct pseudo labels being added. This approach is better because the testing accuracy is increasing and error is decreasing at each iteration.

Table 4: Iterations for Tri-training with disagreement

	CRNN				LSTM				MLP				
Iter	Val acc	P-L	% correct	error	Val acc	P-L	% correct	error	Val acc	P-L	% correct	error	Test acc
0	73.893	0	0.000	0.500	85.191	0	0.000	0.500	76.947	0	0.000	0.500	70.218
1	71.429	376	71.011	0.081	75.212	203	72.906	0.105	76.503	308	78.896	0.080	74.227
2	73.389	236	66.525	0.050	82.486	211	69.668	0.051	85.756	132	67.424	0.064	74.456
3	78.273	344	77.035	0.031	82.698	107	67.290	0.024	87.187	97	68.041	0.046	75.372
4	79.545	197	72.081	0.019	82.051	187	71.123	0.021	90.251	89	59.551	0.058	73.769

Here, P-L is Pseudo-label

4 Evaluation

As observed from Table **??**, the self trained models have higher accuracy than their baselines. Also, tri-trained versions of models have even high accuracy than all baselines and self trained models. As evident from the fact that, tri training use three models to pseudo labels, it is a better approximation than self training.

Table 5: Test Accuracy of Baseline, Self-Training, and Tri-Training Models

	Testing Accuracy	
	Baseline	Self Trained
CNN	67.7	70.22
LSTM	60.94	65.64

Tri-Training	Testing Accuracy
Majority Votes	76.174
Disagreement	75.372

5 Real-time testing

For real time testing of our prescribed models, 5-6 minutes you-tube audios were downloaded for all classes. These .mp3 files were converted into .wav form and cut into 4 second chunks to give approx 100 samples for each class. Thereafter we did the feature extraction and tested on the trained model. Here the predicts were done on the basis of predictions of every model. If two models gave similar results and their probability is more than a threshold then we considered their output as our predicted output. But if no two classes satisfied the above condition, we considered the prediction to be of that model whose probability is maximum.
We had an accuracy of 35%, achieved with the real time audio dataset predicted with majority voting. With disagreement model our accuracy was about 38% which has an overall increment compared to majority voting. This is observed as disagreement model tweaks the model only at its weak points and is better than majority voting. The confusion matrix for both the models are shown in Figure 5 and 6.

6 Generalization of our model for any class

To generalize our models for any given class different than previously trained 10 classes, we employed transfer learning. As shown in Figure 3(Figure 3), in transfer learning, a machine exploits the knowledge gained from a previous task to improve generalization about another. Using this, our model is able to generalize for any class by extracting its layers and feeding it to another new model with very few epochs. Two classes 'Birds' and 'Cats' was extracted for the purpose. Transfer learning on this data revealed a promising accuracy of 80%. Therefore our models are able to generalize on new given classes as well.

References

Below are some references

[1] K. J. Piczak, "Environmental sound classification with convolutional neural networks", in 2015 *IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2015*, **pp. 1–6**

[2] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia, vol. 17, no. 10,* **pp. 1733–1746, 2015.**

[3] Benjamin Elizalde, Ankit Shah, Siddharth Dalmia, Min Hun Lee, Rohan Badlani, Anurag Kumar, Bhiksha Raj, Ian Lane, "An Approach for Self-Training Audio Event Detectors Using Web Data" *IEEE 2017 25th European Signal Processing Conference*

[4] Zhi-Hua Zhou, Ming Li, "Tri-Training: Exploiting Unlabeled Data using Three Classifiers" *IEEE Transactions on Knowledge and Data Engineering*

[5] Bowen Shi, Ming Sun, Chieh-Chi Kao, Viktor Rozgic, Spyros Matsoukas, Chao Wang, "SEMI-SUPERVISED ACOUSTIC EVENT DETECTION BASED ON TRI-TRAINING" *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*

[6] SEBASTIAN RUDER, "An overview of proxy-label approaches for semi-supervised learning" *ruder.io*

[7] Dong-Dong Chen, Wei Wang, Wei Gao , Zhi-Hua Zhou, "Tri-net for Semi-Supervised Deep Learning" *Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*

Appendix

Algorithm 1 Self-training

repeat
  $m \leftarrow train\_model(L)$
  for $x \in U$ **do**
    if $\max m(x) > \tau$ **then**
      $L \leftarrow L \cup \{(x, p(x))\}$
until no more predictions are confident

Figure 1: Self Training

Algorithm 4 Tri-training

for $i \in \{1..3\}$ **do**
  $S_i \leftarrow bootstrap\_sample(L)$
  $m_i \leftarrow train\_model(S_i)$
repeat
  for $i \in \{1..3\}$ **do**
    $L_i \leftarrow \emptyset$
    for $x \in U$ **do**
      if $p_j(x) = p_k(x)(j, k \neq i)$ **then**
        $L_i \leftarrow L_i \cup \{(x, p_j(x))\}$
    $m_i \leftarrow train\_model(L \cup L_i)$
until none of m_i changes
apply majority vote over m_i

Figure 2: Tri Training

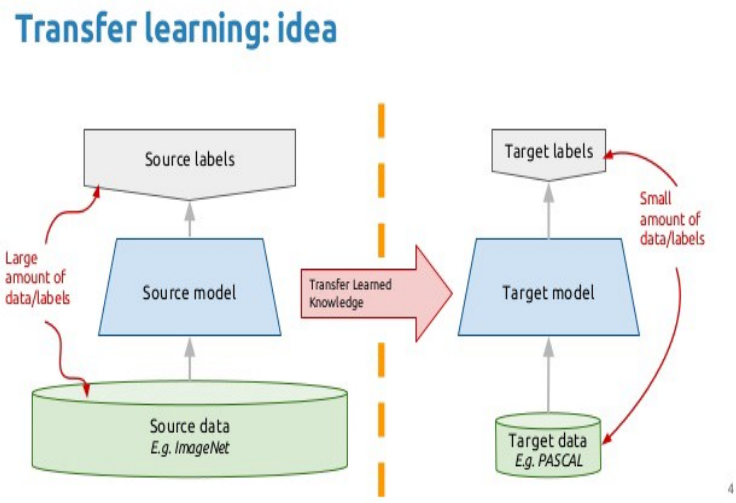

Figure 3: Transfer Learning

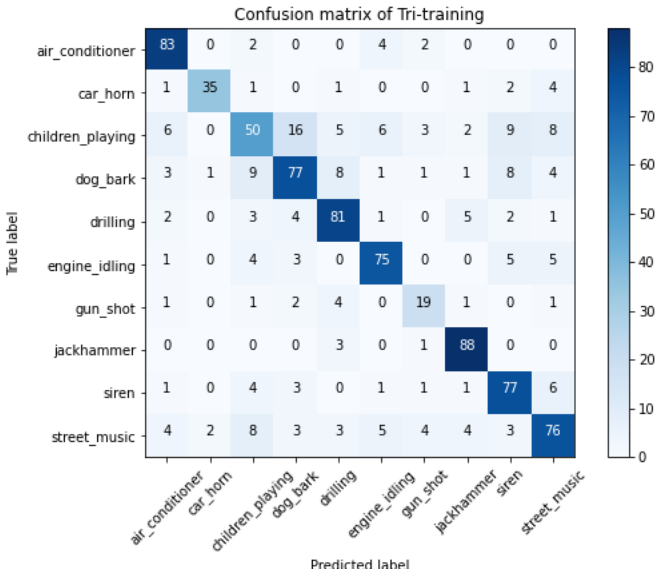

Figure 4: On testing data

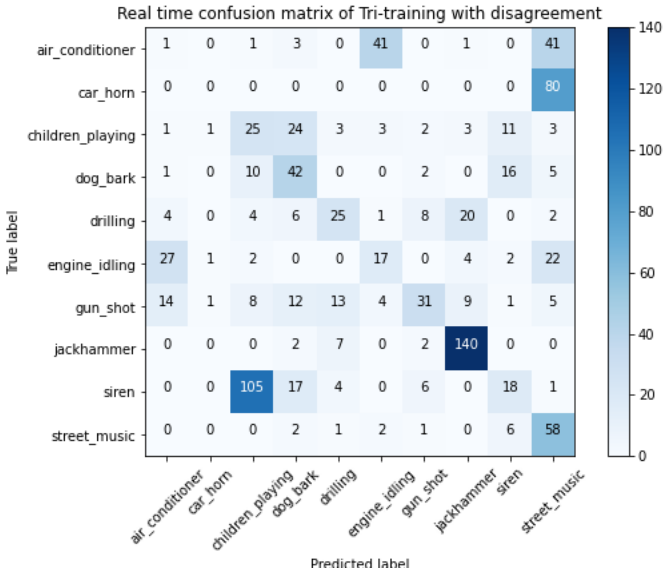

Figure 5: On real-data

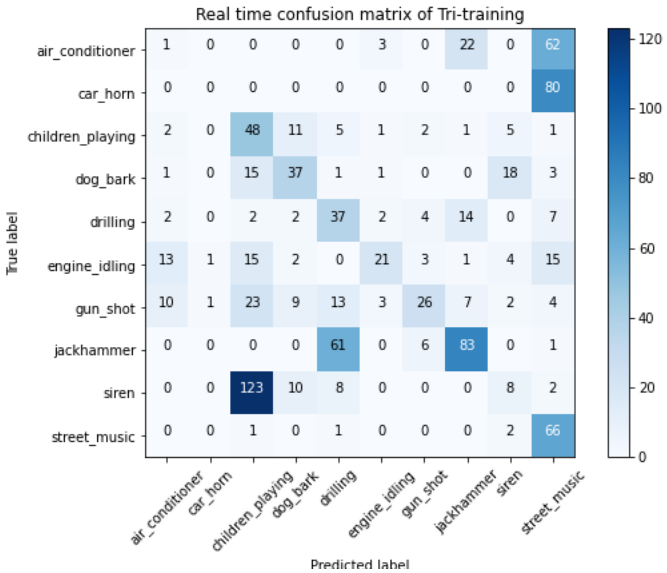

Figure 6: On real-data