EE 621 Speech Processing
Project Report

IMPROVING THE GENERALIZABILITY OF FAKE SPEECH DETECTION SYSTEMS USING DEEP LEARNING APPROACHES

May 2, 2024

G Nikhil Sai
210010018

K Divya Harshitha
200030029

IIT Dharwad

Abstract

The performance of spoofing countermeasure systems depends fundamentally upon the use of sufficiently representative training data. With this usually being limited, current solutions typically lack generalisation to attacks encountered in the wild. For this project we tried to remedy that by testing how training and testing a model on a combination of different datasets works and in the process try to figure out whether it is possible or not and figure out ways to improve. In this report we summarize our experiments and their results mainly based on the AASIST model.

Introduction

A persisting challenge in the design of spoofing countermeasures (CMs) for automatic speaker verification (ASV) is reliability in the face of diverse, unpredictable attacks. Recent studies show that discriminative information (i.e., spoofing artefact) can reside in both spectral and temporal domains. The goal of this project is to understand how a model that is trained on a specific dataset, behaves when given data from a different dataset. We also explore what happens when we train a model on several different datasets. Finally we try to fine tune a pre-trained model with new data from a variety of datasets and see how well it performs.

Project Implementation

The first thing we did as a part of our project was research on the topic and refer to several articles and research papers related to generalization of fake speech detection. A few of the papers referred by us are cited at the end [1][2][3][4][5].

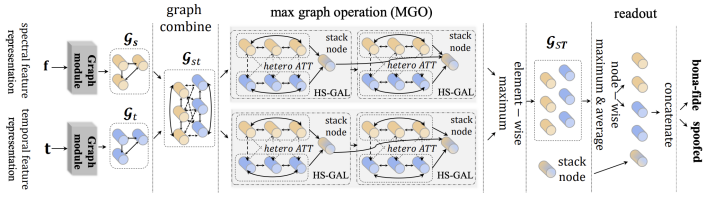

Figure 1: Caption

We replicated the AASIST model and compared the results obtained with the ones stated. After that we trained the large model on a small dataset that contained samples from two speakers each from ASVspoof-2019 and ASVspoof-2015 Datasets. We were able to train the model for 85 epochs due to computational limitations of kaggle.

After this we switched to the smaller model for better training and testing times. Now we combined the data from CMU, LibriTTs and LJ datasets and created a combined dataset. We froze the initial few layers of the model and fine-tuned the model using the above data.

Testing and Experiments

The first experiment was just to replicate the results obtained in the original paper. The training couldn't be completed till the desired 100 epochs due to computational limitations so results were shown only for the model till the point it was able to complete training. The results matched closely with hat was expected.

In the second experiment, we trained the model on a small dataset that we created ourselves. The dataset contained two speakers from ASVspoof-2019 and two from ASVspoof-2015 datasets. We created this dataset by taking the ASVspoof-2019 dataset, then we selected four speakers from it. From the four we chose two and replaced the audio files corresponding to these users with the ones of two users from the ASVspoof-2015 dataset by keeping the file name the same. Then we deleted the data of the remiaining users.

This works because AASIST doesn't use the user data while training or testing, it is just redundant information. Therefore we can change the dataset however we like but we have to make sure that the real and fake speech don't get mixed up.

For the final experiment we used data from CMU, LJ and LIbriTTS datasets along with the ASVspoof-2019 dataset. We combined the datasets similar to how we combined them for the second evaluation. For this experiment we decided to use the smaller model as training would take a very long time otherwise and we had to complete the training within 12 hours due to restrictions on kaggle. We took the pre-trained model, froze the forst few layers as they already have general information about the speech, then we trained the model on the combined dataset for 50 epochs and the results were recorded.

Data Analysis

In the first experiment, we got an EER of 1.02% and min-tCDF of around 0.03%. These results are close to what was expected. They are slightly higher that what was claimed by the paper as the training wasn't complete.

*Continued in the next page

We got the following results for the second experiment

```
CM SYSTEM
        EER             = 6.668615572 % (Equal error rate for countermeasure)

TANDEM
        min-tDCF                = 0.14932589

BREAKDOWN CM SYSTEM
        EER A07         = 0.229994365 % (Equal error rate for A07
        EER A08         = 0.153329576 % (Equal error rate for A08
        EER A09         = 0.000000000 % (Equal error rate for A09
        EER A10         = 1.110289283 % (Equal error rate for A10
        EER A11         = 0.153329576 % (Equal error rate for A11
        EER A12         = 0.306659153 % (Equal error rate for A12
        EER A13         = 0.076664788 % (Equal error rate for A13
        EER A14         = 0.076664788 % (Equal error rate for A14
        EER A15         = 0.229994365 % (Equal error rate for A15
        EER A16         = 0.573635766 % (Equal error rate for A16
        EER A17         = 0.726965342 % (Equal error rate for A17
        EER A18         = 2.220578567 % (Equal error rate for A18
        EER A19         = 0.383323941 % (Equal error rate for A19
```

Figure 2: Caption

We see that both the EER and the min-tCDF are much higher than the original. We can attribute this to two things, a) we didn't take enough training data or b)The model performs worse when data is diverse or both.

We got the following results for the final experiment.

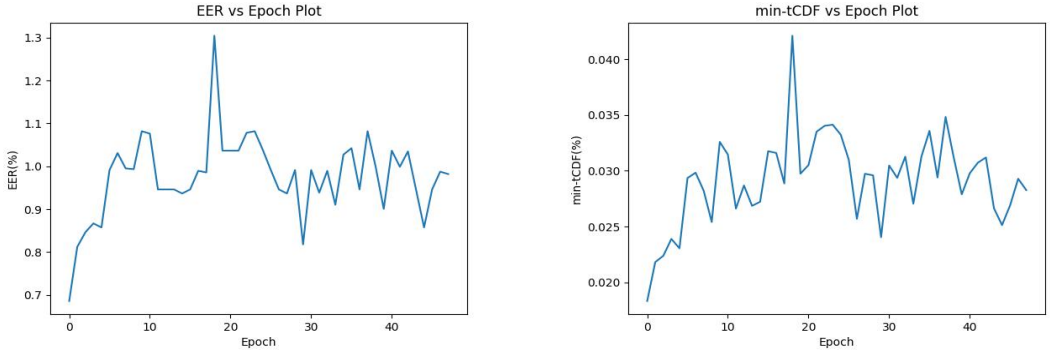

Figure 3: EER and min-tCDF variation across the training epochs

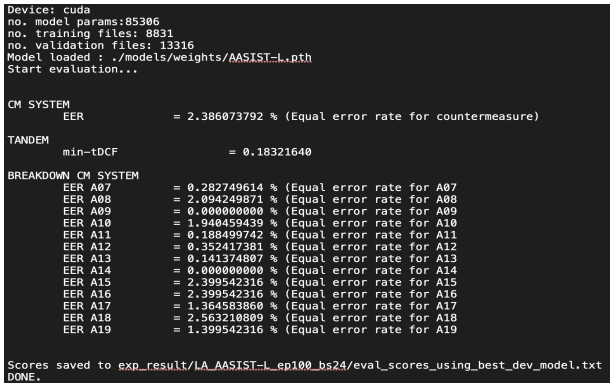

Figure 4: Caption

We observed that the model performed relatively well even for the combined dataset and the accuracy is decent. Freezing the top layers helped in reducing the training time but the model was only able to train for 52 epochs within 12 hours.

Conclusion

From the experiments we were able to conclude that if we want to train the model with multiple datasets, we will need a lot of data for better accuracy. Training the model by freezing the top layer helped us to preserve the high level features of real and fake speech that the previous model learnt. This also helped reduce the training time significantly. We observed that this fine tuning of the pretrained model preserves the data learned by the previous model and makes the model also adapt to the new data. Since we have data from multiple datasets, the model is less likely to learn something specific to a dataset. Therefore we can conclude that the Automatic Speaker Verification systems can be generalized to some extent, if we have a lot of data from different datasets. It also helps if we have a good pretrained model that we can finetune with new data.

Acknowledgment

We thank our advisor, Rishith Sadashiv sir for the guidance and support through out the duration of the project.

References

[1] M. Haoxin, Y. Jiangyan, T. Jianhua, B. Ye, T. Zhengkun, and W. Chenglong, “Continual learning for fake audio detection,” in *Inter-*

speech 2021, ser. interspeech$_2$021. $ISCA, Aug.2021.[Online].Available : http : //dx.doi.org/10.21437/Interspeech.2021 - 794

[2] H. jin Shim, J. weon Jung, and T. Kinnunen, "Multi-dataset co-training with sharpness-aware optimization for audio anti-spoofing," 2023.

[3] H. Tak, M. Todisco, X. Wang, J. weon Jung, J. Yamagishi, and N. Evans, "Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation," 2022.

[4] X. Wang and J. Yamagishi, "Investigating self-supervised front ends for speech spoofing countermeasures," 2022.

[5] Y. Zhang, F. Jiang, and Z. Duan, "One-class learning towards synthetic voice spoofing detection," *IEEE Signal Processing Letters*, vol. 28, p. 937â941, 2021. [Online]. Available: http://dx.doi.org/10.1109/LSP.2021.3076358