Synthetic Speech Detection
Deep Learning Project Final Report

Inna Larina[1], Ilona Basset[1], Folu Obidare[1], Hernán Nenjer[1], and Maksim Komiakov[1]

Skolkovo Institute of Science and Technology, Bol'shoy Bul'var, 30, str. 1, Moscow, 121205

Abstract. This project final report contains a comprehensive account of the developed robust fake speech detection system by leveraging advanced feature extraction techniques and neural network architectures.

Keywords: deepfake · synthetic speech detection · ASV systems

Github repository: Please click here to see our project repo.

1 Introduction

Current work is devoted to the development of a comprehensive fake speech detection system based on state-of-the-art techniques in feature extraction and neural network architectures.

We address the key challenges:

- Growing generation of highly realistic fake speech
- Poor generalization of synthetic speech detection systems

In this paper, we introduce our own fake speech detection system developed based on the framework presented in "Fake Speech Detection Using Residual Network with Transformer Encoder" [5].

2 Dataset

For model training and evaluation it is decided to use ASVspoof2019 LA and ASVspoof2021 DF datasets.

The ASVspoof 2019 LA [3] dataset consists of training, development, and evaluation partitions. The spoofed utterances are generated using 19 VC and TTS algorithms (6 for the training and development sets, 13 for the evaluation set).

The ASVspoof 2021 DF [4] evaluation set is similar to the ASVspoof 2019 LA data, but is intentionally more challenging. It exhibits audio coding and compression artefacts. No new training or development data was released for ASVspoof 2021.

For dataset downloading we implement two classes: ASVspoof2019 and ASVspoof2021 inherited from `datasets.GeneratorBasedBuilder`:

- The `_split_generators()` method downloads and extracts the dataset from the provided URL, and defines the training, validation, and test splits.
- The `_generate_examples()` method reads the metadata file for each split, extracts the relevant information (audio file name, label), and constructs a dictionary for each example containing the audio file path and other metadata.

3 Pipeline

We set ourselves the task of designing the model based on residual network and transformers, training the model on ASVspoof19 dataset and evaluate the model on ASVspoof19 and ASVspoof21 datasets to check the quality and generability of the model.

3.1 Audio Preprocessing

Initial audio files contain records with different duration. To generalize input audio data and enhance the quality of it, we apply the set of preprocessing steps: (1) convert audio to array of amplitudes, (2) resample audio to target frequency (16 kHz or 22.05 kHz), (3) set equal duration for all audios, (4) normalize audio.

3.2 Features extraction

For the training of our model, we use three types of acoustic characteristics extracted from the audio signals: **Log Power Spectrum (LPS)** that analyzes the power distribution across frequencies, **Mel-Frequency Cepstral Coefficients (MFCCs)** that captures the spectral properties across the Mel-scale, closely approximating the human auditory response, **Constant Q Cepstral Coefficients (CQCC)** that utilizes the Constant Q transform to detect fine-grained acoustic manipulations.

These features have proven to be effective in the task of false voice detection in previous works [1], [2].

3.3 Model

We design TE-ResNet [5] model, for more details check the appendix 7. Furthermore, we adopt ResNet18 architecture as baseline solution.

3.4 Metrics

For the evaluation of the model we use the following metrics: balanced accuracy, precision, recall, EER and ROC/AUC. Our choice is justified by target datasets' imbalance.
The EER (Equal Error Rate) is the location on a ROC or DET curve where the false acceptance rate and false rejection rate are equal. In general, the lower the equal error rate value, the higher the accuracy of the model.

3.5 Training

During the training, 3 audio features are extracted using various methods such as MFCC, CQCC and LPS as specified by the user. These characteristics are used individually for training the adopted pretrained ResNet18 model and the model TE-ResNet, and the loss is calculated using the cross-entropy loss function. Subsequently, the weights of the model are updated by back propagation and optimization with Adam.
We have achieved the results, represented in tables 1, 2.

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.025	0.981	0.995	0.996
CQCC	0.079	0.923	0.981	0.986
LPS	0.019	0.981	0.996	0.996

Table 1. Training performance for **ResNet18** on **ASVspoof19** dataset

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.331	0.514	0.898	0.759

Table 2. Training performance for **TE-ResNet** on **ASVspoof19** dataset

3.6 Evaluation

The evaluation of adopted pretrained ResNet18 and trained TE-ResNet model is performed on the test set of ASVspoof19 Dataset and evaluation set of ASVspoof21.

We have achieved the results, represented in tables 3, 4, 5, 6 and figure 1.

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.411	0.893	0.987	0.893
CQCC	0.399	0.695	0.924	0.949
LPS	0.580	0.839	0.988	0.758

Table 3. Evaluation performance for **ResNet18** on **ASVspoof19** dataset

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.333	0.511	0.897	0.7587

Table 4. Evaluation performance for **TE-ResNet** on **ASVspoof19** dataset

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.228	0.801	0.981	0.966
CQCC	0.242	0.656	0.968	0.99
LPS	0.534	0.739	0.982	0.813

Table 5. Evaluation performance for **ResNet18** on **ASVspoof21** dataset

Feature Extractor	Loss	Balanced Accuracy	Precision	Recall
MFCC	0.183	0.626	0.963	1.000

Table 6. Evaluation performance for **TE-ResNet** on **ASVspoof21** dataset

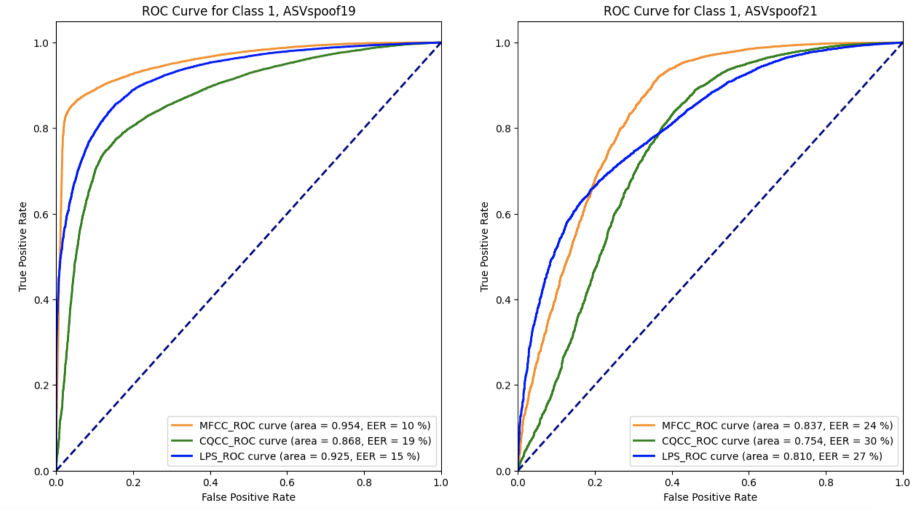

Fig. 1. ROC/AUC, EER for ResNet18

4 Singing Voice Deepfake Detection Challenge

Our fake speech detection model was put to the test in the Singing Voice Deepfake Detection Challenge, where it faced various types of attacks. Although singing voice are much more different data to deal with than regular speech, AI-generated singing voices pose a threat to artists and the music industry. Despite unique challenges due to its musical nature and the presence of strong background music we were still able to achieve about 49% EER. We plan to fur-

ther improve our mobile by combining the baseline for singing voice deepfakes as well as further training on singing voice data to make our model more generable and robust. The table 7 showcases the Equal Error Rates (EER) achieved by our model against each attack type.

Attack Type	EER(%)
Pooled Attack	49.36
Attack A14	43.02
Attack A13	49.06
Attack A12	51.53
Attack A11	51.34

Table 7. EER by TE-ResNet against each attack type

5 Conclusion

In the current work, we propose a fake detection system, which mainly consists of three parts: speech data preprocessing 3.1, 3.2, residual neural network 3.3 and evaluation by metrics 3.4. By following the steps in foundation work [5], we have checked the generability of the model and achieved good outcomes in fake speech detection.
Experimental results on ASVspoof19 and ASVspoof21 demonstrate that ResNet18 is robust model within the target task, although TE-ResNet needs improvement. One possible reason is that ResNet18 is already pretrained, while TE-ResNet isn't pretrained and is lighter than its analogue in the foundation work due to our limited resources.
Besides that, both models ResNet18 and TE-ResNet showed better results on ASVspoof19, on which they were trained, than on ASVspoof21. It is explained by the fact that ASVspoof21 dataset consists of more advanced spoofing attacks. It means that the big drawback remains generability.
And finally, we have compared different approaches for feature extraction. With MFCC we get the best performance of the models, while with CQCC - the worst.

6 Contributions

Inna Larina (45% of work)

- Load ASVspoof19 and ASVspoof21
- Implement code pipeline
- Design TE-ResNet model
- Calculate metrics EER and AUC/ROC

Ilona Basset (5% of work)

- Prepare reports and presentation

Folu Obidare (10% of work)

- Adopt work for competition SVDD2024

Hernán Nenjer (20% of work)

- Feature extraction LPS, CQCC, MFCC
- Train and evaluate ResNet18

Maksim Komiakov (20% of work)

- Design TE-ResNet model and implementation
- Model description

Bibliography

[1] Moustafa Farid Alzantot, Ziqi Wang, and Mani B. Srivastava. Deep residual neural networks for audio spoofing detection. In *Interspeech*, 2019.

[2] Md Sahidullah, Tomi Kinnunen, and Cemal Hanilçi. A comparison of features for synthetic speech detection. 09 2015.

[3] Xin Wang, Junichi Yamagishi, Massimiliano Todisco, Hector Delgado, Andreas Nautsch, Nicholas Evans, Md Sahidullah, Ville Vestman, Tomi Kinnunen, Kong Aik Lee, Lauri Juvela, Paavo Alku, Yu-Huai Peng, Hsin-Te Hwang, Yu Tsao, Hsin-Min Wang, Sebastien Le Maguer, Markus Becker, Fergus Henderson, Rob Clark, Yu Zhang, Quan Wang, Ye Jia, Kai Onuma, Koji Mushika, Takashi Kaneda, Yuan Jiang, Li-Juan Liu, Yi-Chiao Wu, Wen-Chin Huang, Tomoki Toda, Kou Tanaka, Hirokazu Kameoka, Ingmar Steiner, Driss Matrouf, Jean-Francois Bonastre, Avashna Govender, Srikanth Ronanki, Jing-Xuan Zhang, and Zhen-Hua Ling. Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech, 2020.

[4] Junichi Yamagishi, Xin Wang, Massimiliano Todisco, Md Sahidullah, Jose Patino, Andreas Nautsch, Xuechen Liu, Kong Aik Lee, Tomi Kinnunen, Nicholas Evans, and Héctor Delgado. Asvspoof 2021: accelerating progress in spoofed and deepfake speech detection, 2021.

[5] Zhenyu Zhang, Xiaowei Yi, and Xianfeng Zhao. Fake speech detection using residual network with transformer encoder. In *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '21, pages 13–22, New York, NY, USA, 2021. Association for Computing Machinery.

7 Appendix A. TE-ResNet model

The TE-ResNet consists of two parts: transformer encoder and residual network. The transformer encoder is used to pre-process the acoustic features matrix to get the deep feature maps.

TE-ResNet flowchart