Synthetic Speech Classification

AntiSpoofer
SP Cup Team ID: 27612
Independent University, Bangladesh

***Abstract*—The aim of this project is to implement and design a robust synthetic speech classifier for the IEEE Signal Processing Cup 2022 challenge. Here, we learn a synthetic speech attribution model using the speech generated from various text-to-speech (TTS) algorithms as well as unknown TTS algorithms. We experiment with both the classical machine learning methods such as support vector machine, Gaussian mixture model, and deep learning based methods such as ResNet, VGG16, and two shallow end-to-end networks. We observe that deep learning based methods with raw data demonstrate the best performance.**

***Index Terms*—synthetic speech attribution, audio spoofing, end-to-end audio classification.**

I. Introduction

Human-like speech generation from text is no longer a challenge. With the recent advancement in text-to-speech (TTS) algorithms, powerful models such as Tacotron [1], [2], WaveNet [3], and DeepVoice [4] can produce almost natural human voice. This raises the possibility of manipulating someone's voice data [5], which is an alarming issue and can have severe negative impact on the society. Therefore, it is important than ever before to detect such maliciously altered voices. In recent years, many spoof detection challenges had been organized [6]–[8]. Interested readers are referred to [9] and references therein. However, in this challenge, we will explore solutions to a synthetic speech attribution problem, i.e., finding out the type of algorithms generate synthetic speech.

The rest of the paper is structured as follows. In Section II, we discuss the properties of the dataset provided by the SPCUP2022 competition and data preparation technique. In section III, we give an overview of both classical and deep learning approaches that we employ for experimenting on the datasets. In Section IV, we explore how well the features are separated for classification. Subsequently, in Section V, we discuss the experimental results, and in Section VI, we describe the contributions of the team members. Finally, in Section VII we draw concluding remark.

II. Data

In this section, we discuss the training and evaluation data of Part 1 and 2 of the competition as well as the noisy dataset that is used in Part 2 of the competition. We also provide an overview on dataset preparation.

A. Training data

1) Part 1: This dataset is composed of 5000 noise-free audio samples. There are a total of 5 classes, labeled 0 through 4. Each class represents a certain speech synthesizer algorithm that was used to generate the audio samples. Table I illustrates the properties of the audio files from dataset Part 1. However, the identity of the algorithm is hidden. Each class consists of 1000 audio samples in the dataset. However, there is an uneven distribution of male and female voices within the dataset, with some classes having little to no male voices and vice-versa. Audio samples vary in length.

TABLE I: Properties of audio samples from dataset Part 1

Label	No. of samples	Total duration (s)	Mean duration (s)	Std. deviation of duration (s)
0	1000	8255.93	8.26	2.75
1	1000	6426.55	6.43	2.08
2	1000	6359.18	6.36	2.12
3	1000	8136.84	8.14	2.56
4	1000	5615.91	5.62	1.91

2) Part 2: This dataset is composed of 1000 noise-free audio samples, all of which belongs to a single class labeled as "5". This particular label implies that the audio samples were generated using multiple speech synthesizers not among those described in the previous section. The total duration in this case is 6791.28 seconds, mean duration is 6.79 seconds and standard deviation of duration is 2.22 seconds.

B. Evaluation data

This dataset also has two parts. In Part 1, we have 9000 noise-free audio samples with no label information although the samples are generated using audio synthesizing algorithms that are either known or unknown. The average total duration of the dataset is 61539.27 seconds, mean duration is 6.84 seconds and standard deviation of duration is 2.43 seconds. Similarly in Part 2, there are 9000 samples but they have been augmented using different augmentations such as noise injection, adding reverberations and MP3 compression. It is completely unknown as to which tracks have been augmented and how. The labels, average and total duration as well as the standard deviation of duration are similar to Part 1.

C. Noisy data

The second part of the competition emphasizes on synthetic speech attribution in the presence of noisy data. To facilitate this, MATLAB scripts are provided by the competition organizer which can perform the three augmentation techniques on audio samples discussed in the previous section. These augmentation are parametric and the exact values of the parameters during evaluation are not guaranteed to be fixed.

D. Dataset preparation

Since the evaluation dataset has no label information, in order to facilitate the initial training, we adopt stratified, non-overlapping splitting strategy of the provided training data into training, validation and testing sets. When no augmentation is applied, there are 4320 training, 480 validation and 1200 testing samples. However, using augmentation increases the amount of samples three-fold, resulting in 17280, 1920 and 4800 training, validation and testing samples respectively.

III. Methods

In this section, we describe in brief the methodologies that we use for synthetic speech attribution. We explore both classical machine learning approaches and deep learning approaches.

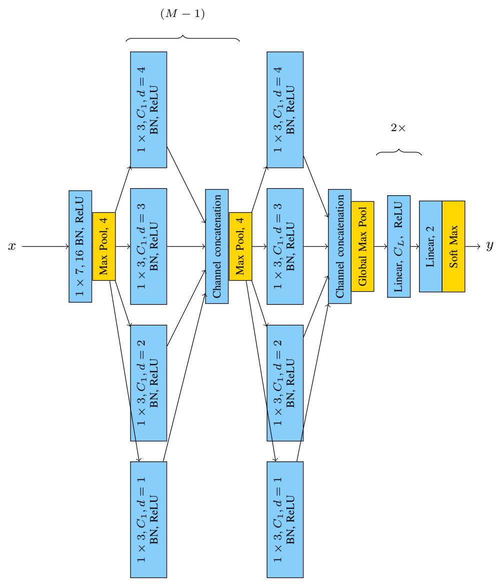
A. Classical machine learning models

1) Support vector machines (SVM): Given a dataset $D = \{\mathbf{d}_i, y_i\}$, where $\mathbf{d}_i \in \mathbb{R}^n$ is an audio sample belonging to class $y_i \in \{0, \ldots, C\}, i = 1, \ldots, N$ and $C \in \mathcal{N}$. We are interested to build a model such that $\mathbf{y} = f(\psi(\mathbf{d}))$, where $\psi(.) : \mathbb{R}^n \mapsto \mathbb{R}^m$. Here, $\psi(.)$ transforms $\mathbf{d}_i$ into a feature vector $\mathbf{x}_i \in \mathbb{R}^m$. To solve the multi-class classification problem, we need $(C - 1)$ 2-class SVM [10] classifiers in one-versus-all setting. That is, the feature $\mathbf{x}_i$ belongs to class C_i or the rest (not C_i) can be defined by the 2-class SVM classifier as $f(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x} + b)$, where $\mathbf{w}$ is the weight vector, b is the intercept term, and $sign(\cdot)$ is the sign function. $f(\mathbf{x})$ takes the value of $+1$ if the sample belongs to class C_i and -1 otherwise. The final decision of a sample belonging to class is taken by majority voting among all the $C - 1$ classifiers.

2) Gaussian mixture model (GMM): In audio signal classification, we can consider that an audio sample belongs to a class parameterized by some distribution such as Gaussian distribution. This motivates us to represent the features $\mathbf{x}$ of the sample as $\mathbf{x} \sim \mathcal{N}(\mu_i, \mathbf{\Sigma}_i)$, where μ_i and $\mathbf{\Sigma}_i$ are the mean and covariance of class y_i. Since the joint probability distribution of a particular class may be multi-modal, i.e. the samples of the class are distributed into multiple clusters, we make a mixture model composed of i multivariate Gaussian distributions, each with mixture probability π_i such that $p(\mathbf{x} \mid \mu, \mathbf{\Sigma}, \pi) = \sum_{i=1}^{C} \pi_i \; p(\mathbf{x} \mid \mu_i, \mathbf{\Sigma}_i)$. There are few approaches to learn this distribution by computing the maximum likelihood. For instance, one can use iterative optimization or expectation maximization techniques (see [11] for details) .

B. Deep learning models

1) Baseline: For establishing our baseline, we consider a shallow and end-to-end time-domain synthetic speech detection net (TSSDNet) proposed by [9], which is inspired from ResNet [12] and Inception-style [13] architectures. Henceforth, we refer to the ResNet-style TSSDNet architecture as Res-TSSDDNet and Inception-style architecture as Inc-TSSDNet. Both use 1D convolutions and batch-normalization, with the Inc-TSSDNet implementing dilated convolution [14] unlike [13], in order to benefit from increased receptive field of



$(M-1)$
$2\times$
x
1×7, 16 BN, ReLU
Max Pool, 4
$1 \times 3, C_1, d = 4$ BN, ReLU
$1 \times 3, C_1, d = 3$ BN, ReLU
$1 \times 3, C_1, d = 2$ BN, ReLU
$1 \times 3, C_1, d = 1$ BN, ReLU
Channel concatenation
Max Pool, 4
$1 \times 3, C_1, d = 4$ BN, ReLU
$1 \times 3, C_1, d = 3$ BN, ReLU
$1 \times 3, C_1, d = 2$ BN, ReLU
$1 \times 3, C_1, d = 1$ BN, ReLU
Channel concatenation
Global Max Pool
Linear, C_L, ReLU
Linear, 2
Soft Max
y

Fig. 1: The Inception-style TSSDNet architecture (adapted from [9]).

the model. We use the same architecture except the final layer, which we modify to accommodate the larger number of class labels, i.e. 5 without "unknown" label and 6 with "unknown" label, in contrast to the original work.

Both models accept raw audio signal as inputs, which we keep to be exactly 6 seconds in length during training and testing by utilizing either repetition (for shorter length waveform or trimming (for waveform with longer length than 6 seconds). The architecture of Inc-TSSDNet is illustrated in Figure 1.

2) Convolutional deep learning methods: For exploring more possibilities, we consider three convolutional neural networks: ResNet34, ResNet18 and VGG16 (see [15] for more discussion). All the methods uses a stack of convolutional layers containing filters of size 3x3 followed by fully connected layers [16]. We incorporate batch normalization with VGG16 due to the poor performance of the standard VGG16. To build speech synthesis classifiers using these algorithms requires the raw data to be transformed to features, which can be done by using feature extraction methods, such as mel-frequency cepstral coefficients (MFCCs) [17].

IV. Feature Analysis

Figure 2 shows the t-SNE [18] plots of the feature embeddings for the methods discussed above. For the Res-TSSDNet and Inc-TSSDNet architectures, we take the output of the final linear layer just before the classification layer. These are 32-dimensional vectors. By contrast, mel-spectogram cepstral coefficients (MFCC) features were computed from raw waveform directly. In both cases, principal component analysis (PCA) [19] was applied for dimensionality reduction, and subsequently the t-SNE plots were computed. It is seen that Inc-TSSDNet achieves the maximum inter-class separation while preserving intra-class compactness regardless of whether

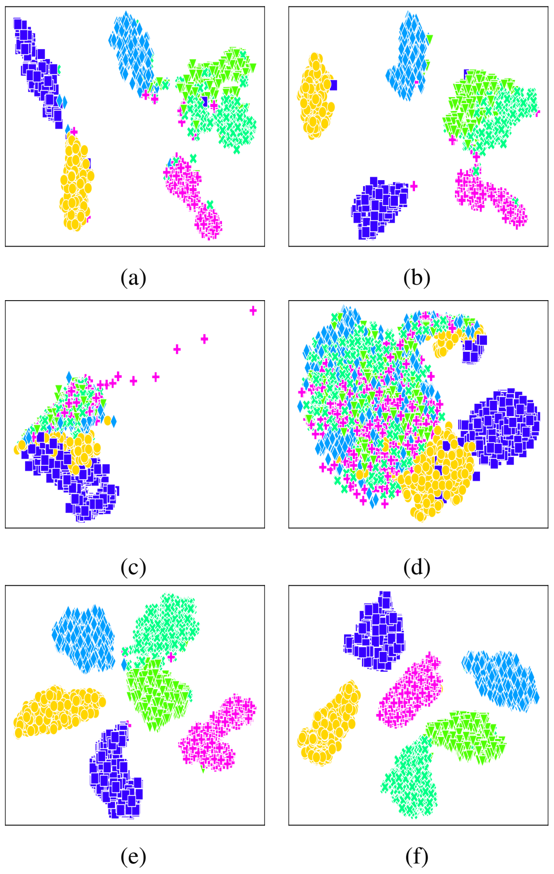


Fig. 2: t-SNE plots of the feature embeddings for (a) Res-TSSDNet, no augmentations, (b) Inc-TSSDNet, no augmentations, (c) MFCC, with augmentation (d) VGG16, with augmentations, (e) Res-TSSDNet, with augmentations, and (f) Inc-TSSDNet, with augmentations. The colors blue, yellow, light blue, green, light green and pink represent class labels $\{0, 1, 2, 3, 4, 5\}$, respectively.

augmented data is used. In contrast, MFCC fails to provide enough discriminative features for reliable classification.

V. Experimental Results

In this section, we describe the experimentation setup and report results for the architectures we have discussed in the previous section.

A. Experimental setup and hyperparameters

All the models were trained on a server machine running a 16-core Intel® Xeon® CPU E5-2640 v3 @ 2.60GHz, 8 Nvidia Tesla K80 GPUs with 12 GB virtual memory. For implementing the classical machine learning models, we use the scikit-learn [20] library, while for implementing the deep learning based models, we use PyTorch with PyTorch Lightning [21] library. The implementations are available at GitHub[1]. We train all the models except GMM with 200 epochs with a batch size of 128 and an initial learning rate of 10^{-3}. We use exponential learning rate scheduler with a learning rate decay factor $\gamma = 0.95$ per epoch. For TSSDNet type models, we used raw data as features, where as, with other models, we used different features listed in Table II.

[1]https://github.com/AGenCyLab/SPCUP2022

TABLE II: Comparison of different deep learning methods. The accuracies reported are on the held-out testing set of the provided training data as discussed in Section II. The features are mel-spectrogram (MS), mel-frequency cepstral coefficients (MFCC), and raw waveform (RW).

Methods	Features	Accuracy	
		Unaugmented data	Augmented data
ResNet34	MS	0.95	0.97
ResNet18	MS	**0.98**	0.97
VGG16	MS	0.70	0.68
Inc-TSSDNet	RW	0.96	**1.00**
Res-TSSDNet	RW	0.93	0.99
SVM	MFCC	0.86	0.97
GMM	MFCC	0.25	0.39

B. Speech synthesis classifiers

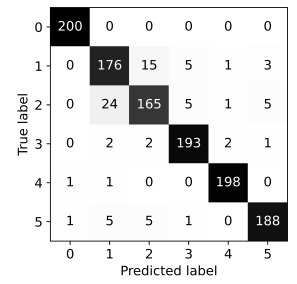
Figure 3 shows the confusion matrices of various classifiers that we described in Section III on samples belonging to the held-out test set. It can be noted that after using augmented data, TSSDNet type models perform really well during testing. As Figure 2 shows, this is possible due to the discriminative features learnt by the networks. Table II summarizes the performance of the models.

VI. Team members and contributions

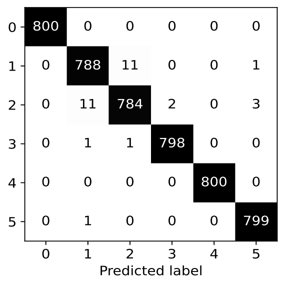
Mahieyin Rahmun contributed in planning the experiment, implementing deep learning based models, and writing. Tanjim Taharat Aurpa contributed in data and feature analysis as well as writing. Rafat Hasan Khan contributed in implementing deep learning based models and writing. Sadia Khan and Zulker Nayeen Nahiyan contributed in writing. Mir Sayad Bin Almas and Rakibul Hasan Rajib contributed in reviewing. Sakira Hassan contributed in reviewing the code and the report.

VII. Discussion and Conclusion

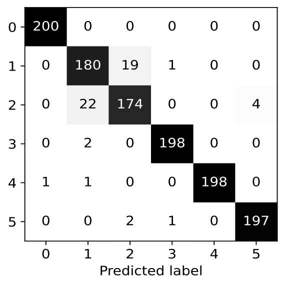
From Table II, we observe that Inc-TSSDNet performs the best on augmented data with raw audio, while ResNet18 performs best on unaugmented data with mel-spectrogram features. Both Inc-TSSDNet and Res-TSSDNet see improvements in their performances with augmented data. We argue that both the larger number of training samples along with the numerous variations of the same audio contribute towards helping the models learn a robust feature space required for better classification. This is also observed when SVM classifier is trained with MFCC features. VGG16 has the worst performance among all, since it only considers mel-spectrogram features and does not have all required information to distinguish the classes. However, if we use a denser network such as ResNet34 (accuracy values are 0.95 and 0.97 for unaugmented and augmented data, respectively), the performance of the classifier improves. Based on our observations, we have decided to put



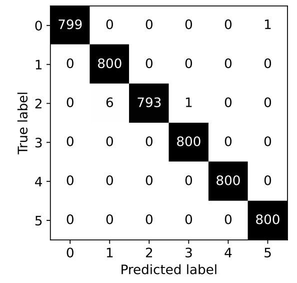
(a)



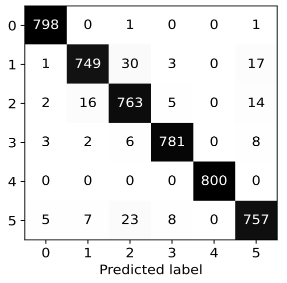
(b)



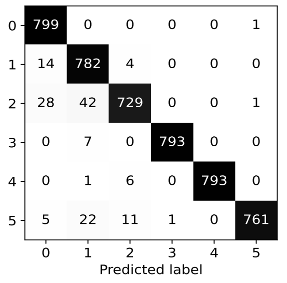
(c)



(d)



(e)



(f)

Fig. 3: Confusion matrices for (a) Res-TSSDNet, no augmented data, (b) Res-TSSDNet, augmented data, (c) Inc-TSSDNet, no augmented data, (d) Inc-TSSDNet, augmented data, (e) ResNet34, augmented data, and (f) SVM, augmented data.

forward Inc-TSSDNet as the detector of choice for synthetic speech attribution. According to Codalab[2], on a subset of the evaluation set part 1, the model achieves an accuracy of 0.84, while scoring 0.91 on evaluation set part 2, which is the highest among all the other models presented in this paper.

Acknowledgement

Thanks to AGenCy Lab[3] for the computational utilities.

References

[1] Y. Wang, R. Skerry-Ryan, D. Stanton *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *INTERSPEECH: Conference of the International Speech Communication Association*, 2017.

[2] R. Skerry-Ryan, E. Battenberg, Y. Xiao *et al.*, "Towards end-to-end prosody transfer for expressive speech synthesis with Tacotron," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4693–4702.

[3] A. van den Oord, S. Dieleman, H. Zen *et al.*, "WaveNet: A Generative Model for Raw Audio," in *Proc. 9th ISCA Workshop on Speech Synthesis Workshop*, 2016, p. 125.

[4] S. Ö. Arık, M. Chrzanowski, A. Coates *et al.*, "Deep Voice: Real-time neural text-to-speech," in *International Conference on Machine Learning*. PMLR, 2017, pp. 195–204.

[5] J. Fletcher, "Deepfakes, artificial intelligence, and some kind of Dystopia: The new faces of online post-fact performance," *Theatre Journal*, vol. 70, no. 4, pp. 455–471, 2018.

[6] Z. Wu, T. Kinnunen, N. Evans *et al.*, "ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *16th Annual Conference of the International Speech Communication Association*, 2015.

[7] Z. Wu, J. Yamagishi, T. Kinnunen, C. Hanilçi *et al.*, "ASVspoof: the automatic speaker verification spoofing and countermeasures challenge," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 588–604, 2017.

[8] M. Todisco, X. Wang, V. Vestman *et al.*, "ASVspoof 2019: Future horizons in spoofed and fake audio detection," in *Proceedings of INTERSPEECH*, 2019.

[9] G. Hua, A. B. J. Teoh, and H. Zhang, "Towards end-to-end synthetic speech detection," *IEEE Signal Processing Letters*, vol. 28, pp. 1265–1269, 2021.

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.

[11] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[13] C. Szegedy, W. Liu, Y. Jia *et al.*, "Going deeper with convolutions," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[14] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *4th International Conference on Learning Representations, ICLR*, 2016.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations*, 2015.

[17] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of MFCC," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[18] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

[19] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] W. Falcon and The PyTorch Lightning team, "PyTorch Lightning," 3 2019. [Online]. Available: https://github.com/PyTorchLightning/pytorch-lightning

[2]https://codalab.lisn.upsaclay.fr

[3]https://agencylab.github.io/