ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095043

# ASSD: SYNTHETIC SPEECH DETECTION IN THE AAC COMPRESSED DOMAIN

*Amit Kumar Singh Yadav*[†], *Ziyue Xiang*[†], *Emily R. Bartusiak*[†],
*Paolo Bestagini*[‡], *Stefano Tubaro*[‡], *Edward J. Delp*[†]

[†] Video and Image Processing Lab (VIPER), School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana, USA

[‡] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy

## ABSTRACT

Synthetic human speech signals have become very easy to generate given modern text-to-speech methods. When these signals are shared on social media they are often compressed using the Advanced Audio Coding (AAC) standard. Our goal is to study if a small set of coding metadata contained in the AAC compressed bit stream is sufficient to detect synthetic speech. This would avoid decompressing of the speech signals before analysis. We call our proposed method AAC Synthetic Speech Detection (ASSD). ASSD extracts information from the AAC compressed bit stream without decompressing the speech signal. ASSD analyzes the information using a transformer neural network. In our experiments, we compressed the ASVspoof2019 dataset according to the AAC standard using different data rates. We compared the performance of ASSD to a time domain based and a spectrogram based synthetic speech detection methods. We evaluated ASSD on approximately 71k compressed speech signals. The results show that our proposed method typically only requires 1000 bits per speech block/frame from the AAC compressed bit stream to detect synthetic speech. This is much lower than other reported methods. Our method also had a 9.7 percentage points higher detection accuracy compared to existing methods.

***Index Terms***— Synthetic speech detection, speech forensics, compressed speech, metadata, transformer

## 1. INTRODUCTION

Synthetic speech is speech signals generated using computers to sound like human speech [1]. Traditionally, Markov models were used for speech synthesis [2]. Modern deep learning based methods [3]–[7] have replaced traditional methods because they generate high quality and semantically consistent speech that is almost indistinguishable from pristine speech (*i.e.*, speech recorded from human speakers).

This perceptually narrowed gap between synthetic and authentic human speech has been a driver for many applications, such as virtual assistants, online learning, and entertainment. However, these methods also enable compelling impersonation [8], voice cloning [9], and voice conversion [10] attacks. In 2021, an impersonator using such realistic synthesized speech targeted a financial transaction of $40 million with Goldman Sachs [11]. It is important to detect synthesized speech to prevent such attacks.

This material is based on research sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under agreement number FA8750-20-2-1004. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, AFRL or the U.S. Government. Address all correspondence to Edward J. Delp, ace@ecn.purdue.edu.

Many methods have been proposed for synthetic speech detection. These methods are mainly based on three types of approaches: hand-crafted features (*e.g.*, cepstral coefficients) [12]–[14], spectrograms [15], or time-domain speech analysis [16]. Most of these methods are trained and evaluated on uncompressed speech signals. They suffer a performance drop when evaluated on compressed speech signals and consider compression as a problem. Here, we address this paradigm of considering compression as a problem and show that the coding metadata information in the compressed bit stream is in itself an asset for forensic analysis of speech signal. Advanced Audio Coding (AAC), is one of the most popular lossy audio compression methods [17], [18]. It introduces distortion or compression artifacts in the decompressed signal. In this paper, we investigate if the coding metadata information contained in the AAC compressed bit stream is sufficient to detect compressed synthetic speech. Our proposed method does not require the decompressed speech signal.

The main contributions of this paper are: a) we investigate the possibility of using only coding metadata extracted from the compressed bit stream to detect synthetic speech; b) we propose AAC Synthetic Speech Detection (ASSD) that is based on transformer neural networks and uses AAC coding metadata for synthetic speech detection; c) we show that ASSD can outperform existing methods and typically only requires 1 kilobit (kb) of information from each speech block/frame. Overall, this shows that coding metadata information in the compressed domain can be exploited for efficient synthetic speech detection. The metadata information can be jointly used with time-domain audio waveform to improve existing methods, thus turning the compression problem into an asset.

## 2. BACKGROUND

### 2.1. Related Work

Most approaches for uncompressed synthetic speech detection use hand-crafted temporal and spectral features with Support Vector Machines (SVMs), neural networks, or Gaussian Mixture Models (GMMs) [19]–[23]. Common hand-crafted features include Linear Frequency Cepstral Coefficients (LFCCs) [19], Constant Q Transform (CQT) [19], Mel Frequency Cepstral Coefficients (MFCCs) [20], Constant Q Cepstral Coefficients (CQCCs) [23], and Cochlear Filter Cepstral Coefficients (CFCCs) [22]. More details about synthetic speech detection features are described in [14], [24]. Recently, spectrogram-based methods have outperformed hand crafted approaches [15], [25]. A spectrogram is a 2D representation of an audio signal where the horizontal axis represents time and the vertical axis represents frequency bands [26], [27]. Spectrogram approaches typically use either Convolutional Neural Networks (CNNs) [15] or transformers [15], [25] to detect synthetic speech. Alternative techniques use the time-domain signal. For example, Time-Domain Synthetic Speech Detection Net (TSSDNet) [16] is a CNN that directly

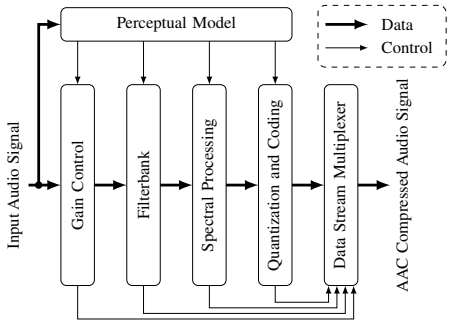


**Fig. 1**. The block diagram of a basic AAC encoder.

uses time-domain speech signal to detect synthetic speech.

While there are some reported work using information extracted from the compressed domain for forensic applications, they are not related to synthetic speech detection. For example, in [28], [29] MPEG-1/2 Audio Layer 3 (MP3) [30] encoding parameters are used to detect multiple audio compression. In [31], scale factors (SF) are used to detect double Advanced Audio Coding (AAC) [17] compression. In this paper, we use transformer networks [32] for forensic analysis. Transformers have been widely used in natural language processing and more recently have been used in signal and image processing applications [33]–[36]. Our transformer network is based on [35], [37], [38], which was originally designed for general audio classification (*e.g.*, identifying animal noises, music).

### 2.2. AAC Audio Compression

Advanced Audio Coding (AAC) [18], [39] is an audio compression standard that is the successor of the MPEG-1/2 Audio Layer 3 (MP3) [39] standard. Compared to MP3, AAC audio compression has higher coding efficiency and audio quality [39]. There are many configurations in AAC, known as profiles, and each profile can introduce new coding tools [18]. Commonly used AAC profiles include `AAC-LC`, `AAC-Main`, `AAC-SSR`, `HE-AAC`, and `HE-AAC v2` [18], [39]. In this paper, we focus on the forensic analysis of `HE-AAC` speech signals as it is one of the latest AAC configurations for compressing monophonic speech signals [18]. The information that we extract from the compressed domain is also common to many other AAC profiles.

Fig. 1 shows the block diagram of a basic AAC encoder. The input to an AAC encoder is a time-domain audio signal. The input signal is partitioned into temporal blocks/frames of 2048 samples with a shift of 1024 samples. The AAC encoder compresses the entire audio signal by processing each block sequentially. The time samples of each block are processed by the Perceptual Model and the Gain Control [17], [18]. The Perceptual Model is based on psychoacoustic analysis of the Human Hearing System (HHS) [40]. The Gain Control is only used in the `AAC-SSR` profile, so it is not used in our experiments. The time-domain samples from each block are directly passed to the Filterbank. In the Filterbank, the samples are windowed and transformed using Modified Discrete Cosine Transform (MDCT) [17], [18]. The Perceptual Model will choose between two windowing schemes: `long` and `short`[1]. `long` preserves higher spectral resolution, and `short` preserves higher time resolution. If `long` windowing scheme is selected, then a window of width 2048 is used for windowing; otherwise, eight non-overlapping windows of width 256 are used. After windowing, MDCT is used on each window. For both `long` and `short` schemes, the output of the Filterbank is a vector of 1024 spectral coefficients. The output spectral coefficients from the Spectral Processing step are used in Quantization and Coding. For quantization, the spectral coefficients are first grouped into scale factor bands, where all coefficients in a band share one scale factor[18]. For each spectral coefficient, $x$, is associated to a scale factor and is quantized as

$$q(x) = \text{sign}(x) \cdot \text{round}\left[\left(|x|/\sqrt[4]{2}^{\text{SF}}\right)^{3/4} - \alpha\right], \tag{1}$$

where SF is the scale factor and $\alpha$ is a small constant. After quantization, the spectral coefficients are compressed with Huffman coding [17], [18]. There are 12 predefined Huffman codebooks in AAC [17], [18]. The coded spectral coefficients as well as control information from all steps are sent to the Data Stream Multiplexer (DSM) step to generate the AAC compressed bit stream.

[1]For simplicity, we categorize windowing schemes by the window width and therefore ignore the transitioning windows between `long` and `short` since their width is also 2048.

## 3. PROPOSED APPROACH

Our goal is to detect synthetic speech using information we extract from the AAC compressed bit stream. We denote an AAC bit stream by $\mathcal{X}$ (*i.e.*, the output of an AAC encoder). $\mathcal{X}$ can be thought of as a sequence of AAC compressed audio signal blocks/frames, where each block corresponds to the result of 2048 time-domain speech samples being AAC compressed. Let us consider an AAC compressed speech signal that contains the first $L$ blocks. That is, $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L\}$, where $\boldsymbol{x}_i$ is the $i$-th block in the speech signal and $L$ is the number of blocks. For the $i$-th block $\boldsymbol{x}_i$, we extract an $N$-dimensional feature vector $\boldsymbol{m}_i \in \mathbb{R}^N$. Hence, given any speech signal $\mathcal{X}$ we have a corresponding representation $\mathcal{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_L\}$. Our goal is to use the representation $\mathcal{M}$ to estimate a label $y$ indicating if $\mathcal{X}$ is a synthetic speech signal or real human speech.

The block diagram of our proposed ASSD method is shown in Fig. 2. ASSD analyzes $L$ AAC compressed blocks $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L\}$. For the $i$-th block $\boldsymbol{x}_i$, we extract the information from the AAC compressed block as defined in Table 1. The information is related to temporal block location, windowing scheme, MDCT coefficients, scale factors, Huffman table indices, and sampling rate. Note we are not decompressing the signal. We chose these parameters because they are common to many of the AAC profiles.

The information in $\boldsymbol{x}_i$ is concatenated into a feature vector $\boldsymbol{m}_i$, where $\boldsymbol{m}_i \in \mathbb{R}^N$. $N$ depends on the choice of the information from the compressed bit stream used for analysis, which is discussed in Sec. 4.4. We use block grouping on $\mathcal{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_L\}$ to obtain $\mathcal{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_{L'}\}$. Each element in $\mathcal{B}$ is formed by overlapping $bshape$ blocks from $\mathcal{M}$ with a stride of $bstride$. For example, $\boldsymbol{b}_1$ is formed by grouping $\boldsymbol{m}_1$ to $\boldsymbol{m}_{bshape}$ blocks and the next element $\boldsymbol{b}_2$ is formed by grouping $bshape$ blocks starting from $\boldsymbol{m}_{1+bstride}$. Therefore, $L' = \lfloor (L-bshape)/bstride \rfloor + 1$. Block

**Table 1**. Information extracted from the AAC compressed bit stream that is used in our method. Details about each parameter can be found in [17], [18]

| Parameter | Description |
|---|---|
| `frame_indx` | Temporal location of the block |
| `WindowSequence` | Windowing Scheme (WS) |
| `pSpec` | Contains MDCT coefficients |
| `L` | Number of MDCT coefficients in each window |
| `GlobalGain, pScaleFactor` | SF info (size 41 or 61) |
| `pCodeBook` | Huffman table selection info |
| `aacdec_sample_rate` | Sampling rate |

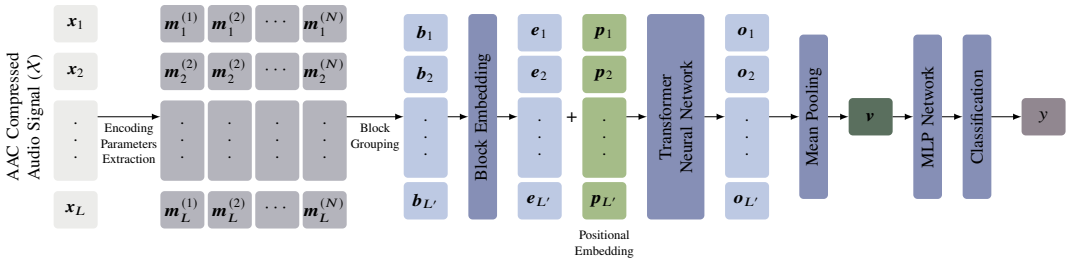


**Fig. 2**. The block diagram of our proposed method. $\boldsymbol{m}_i^{(j)}$ represents the $j$-th element of $\boldsymbol{m}_i$.

grouping improves the performance of ASSD. The block grouping configuration is discussed in Sec. 4.4. For each $\boldsymbol{b}_i \in \mathcal{B}$, we find a block embedding $\boldsymbol{e}_i$ using a linear projection and add a positional embedding vector $\boldsymbol{p}_i$ which is updated during training. Using the transformer neural network [32] adapted from Self-Supervised Audio Spectrogram Transformer (SSAST) [35], we process each vector (*i.e.*, $\boldsymbol{e}_i + \boldsymbol{p}_i$) to obtain a 728-dimensional representation $\boldsymbol{o}_i$ corresponding to each block $\boldsymbol{b}_i$ (Fig. 2). We use mean pooling to compute the mean of all $\boldsymbol{o}_i$'s, which results in the vector $\boldsymbol{v}$. Finally, we process $\boldsymbol{v}$ using a Multi Layer Perceptron (MLP) network to obtain a classification label $y$. The MLP network consists of a normalization layer, a linear layer with 728 neurons, and a final output layer with 2 neurons using softmax activation (Fig. 2). The loss function $\mathcal{L}$ used in ASSD is the sum of Binary Cross Entropy (BCE) loss $\mathcal{L}_{\text{BCE}}$ [35] and center loss $\mathcal{L}_{\text{center}}$ [41]. The BCE loss is used for binary classification. The center loss can improve the discriminative power by minimizing the intra-class variations of $\boldsymbol{v}$ for both synthetic and pristine speech classes.

## 4. EXPERIMENTAL RESULTS

In this section, we present the datasets used in our experiments, the training strategy of AAC Synthetic Speech Detection (ASSD), and the experimental results. We also describe our ablation study.

### 4.1. Dataset

The datasets used in our experiments are based on the Logical Access (LA) part of the ASVspoof2019 dataset [42], [43], which contains 121,461 uncompressed pristine and synthetic speech signals. The ASVspoof2019 dataset has been split into the training set $D_{tr}$, validation set $D_{dev}$, and evaluation set $D_{eval}$ with an approximate ratio of 1:1:3. Each set has approximately 89% synthesized and 11% pristine speech samples, making the dataset highly unbalanced. The synthesized speech samples are generated using neural networks, vocoders, and waveform concatenation as described in [43], while the pristine speech samples are recorded from human speakers. The pristine speech samples in the three splits are disjoint in terms of speakers. There are in total 63.9k synthesized speech samples in $D_{eval}$, where approximately 61.5k samples are generated from synthetic speech generation methods that do not coincide with $D_{dev}$ or $D_{tr}$. All the speech signals in ASVspoof2019 are monophonic and stored using the Free Lossless Audio Codec (FLAC) format.

To generate our experimental datasets, we compressed each of the speech signals in ASVspoof2019 using `HE-AAC` at a data rate of 128kbps, which is commonly used by social networks [44]. In our evaluation, we also used AAC compressed $D_{eval}$ at data rates of 16kbps, 32kbps, and 64kbps to investigate how our proposed approach extends to more aggressively compressed speech signals.

### 4.2. Training Strategy

To train ASSD, we use Adam [45] optimizer for 100 epochs with a batch size of 48. The initial learning rate is set to $10^{-3}$, and it is updated using an adaptive learning schedule. The ASVspoof2019 dataset [42] is highly unbalanced. To cope with the skewed label distribution, at each training step, we sample equal number of data points for each label. This is known as balanced sampling. We train and validate ASSD on AAC compressed $D_{tr}$ and $D_{dev}$ sets at 128kbps and select the version of ASSD with best accuracy on $D_{dev}$ for evaluation. We then evaluate on $D_{eval}$ compressed with data rates of 16kbps, 32kbps, 64kbps, and 128kbps.

**Table 2**. Experiment #1 Results. Performance of CCT [15] and TSSDNet [16] trained on uncompressed $D_{train}$ set and evaluated on both uncompressed and AAC compressed (at constant data rate of 128kbps) $D_{eval}$ set.

| Method | Evaluation Dataset | Balanced Accuracy (%) | Accuracy (%) | AUPRC (%) |
|---|---|---|---|---|
| **TSSDNet** | Uncompressed $D_{eval}$ | 96.27 | 95.17 | 99.84 |
| | Compressed $D_{eval}$ | 73.82 | 54.29 | 97.36 |
| **CCT** | Uncompressed $D_{eval}$ | 94.13 | 94.98 | 68.76 |
| | Compressed $D_{eval}$ | 64.72 | 88.62 | 36.30 |

**Table 3**. Experiment #2 Results. Performance comparison between ASSD (proposed), CCT [15] and TSSDNet [16]. Class 0 and Class 1 refer to pristine and synthetic, respectively.

| Data Rate | Method | Class 0 Accuracy (%) | Class 1 Accuracy (%) | Balanced Accuracy (%) | Accuracy (%) | AUPRC (%) | Input Vector Size |
|---|---|---|---|---|---|---|---|
| **128kbps** | TSSDNet | 93.32 | 72.70 | 83.01 | 74.83 | 98.94 | 96,000 |
| | CCT | 59.00 | 96.80 | 77.90 | 92.90 | 68.70 | 65,536 |
| | **ASSD** | 80.86 | 84.96 | 82.91 | 84.53 | 98.94 | 1,860 |
| **64kbps** | TSSDNet | 93.32 | 72.70 | 83.01 | 74.83 | 98.94 | 96,000 |
| | CCT | 59.00 | 96.80 | 77.90 | 92.90 | 68.70 | 65,536 |
| | **ASSD** | 80.86 | 84.96 | 82.91 | 84.53 | 98.94 | 1,860 |
| **32kbps** | TSSDNet | 93.22 | 72.83 | 83.02 | 74.94 | 98.94 | 96,000 |
| | CCT | 59.07 | 96.81 | 77.94 | 92.91 | 68.39 | 65,536 |
| | **ASSD** | 80.69 | 84.10 | 82.40 | 83.75 | 98.89 | 1,860 |
| **16kbps** | TSSDNet | 90.27 | 73.35 | 81.81 | 75.10 | 98.77 | 96,000 |
| | CCT | 50.30 | 97.32 | 73.80 | 92.47 | 63.85 | 65,536 |
| | **ASSD** | 75.91 | 86.98 | 81.44 | 85.84 | 98.82 | 1,860 |

### 4.3. Results

We use accuracy [46], balanced accuracy [47] and Area Under Precision Recall Curve (AUPRC) [48] as performance metrics for our experiments. The Class 0 Accuracy = $TN/(TN+FN)$ and Class 1 Accuracy = $TP/(TP+FP)$ ,where $TP$ is True Positives, $FP$ is False Positives, $TN$ is True Negatives, and $FN$ is False Negatives.

**Experiment #1.** The goal of this experiment is to show that AAC compression can reduce the performance of synthetic speech detection

methods. We examined the performance of two existing methods: a time-domain method Time-Domain Synthetic Speech Detection Net (TSSDNet) [16] and a spectrogram method Compact Convolutional Transformer (CCT) [15] on uncompressed speech signals and speech decompressed from 128kbps AAC compression (we refer to them as compressed speech). We used TSSDNet as it is one of the most promising synthetic speech detection methods that outperformed hand-crafted feature-based methods. We trained TSSDNet and CCT on uncompressed speech and then evaluated their performance on both uncompressed speech and compressed speech. The results are shown in Table 2. Note that the accuracy of TSSDNet dropped 41 percentage points , and the balanced accuracy of CCT dropped 29 percentage points when being evaluated on compressed speech.

**Experiment #2.** We examined the performance of ASSD, and compared it with TSSDNet and CCT. Note that TSSDNet and CCT require the complete decompressed speech signal where ASSD requires only information from the compressed bit stream. Each of the parameter extracted from the compressed bit stream (Table 1) has distinct level of importance in synthetic speech detection. In the ablation study reported below in Sec. 4.4, we discovered that the most relevant parameters are the temporal location or index of the block/frame (*i.e.*, `frame_indx`) and the information related to scale factors (*i.e.*, `GlobalGain`, `pScaleFactor`), which contain 62 scalar parameters per block corresponding to 1kb. Note that a compressed block contains information corresponding to ~33kb [17], [18]. ASSD analyzes the first $L$ blocks/frames in the speech signal to make a decision. This corresponds to 1860 scalar parameters or 30kb of data. We used 128kbps compressed bit stream for training and validation. For evaluation, we used compressed speech at 128kbps, 64kbps, 32kbps, and 16kbps. TSSDNet and CCT were also retrained using compressed speech at 128kps. The input vector size and experimental results for each method are shown in Table 3. It can be seen that CCT performed well in detecting synthetic speech, however, its performance on pristine speech (*i.e.*, Class 0 Accuracy) is almost random. The balanced accuracy and AUPRC of CCT are on average 5.5 percentage points and 31.5 percentage points less than that of ASSD. TSSDNet performed well on pristine speech, however, the overall detection accuracy is on average 9.7 percentage points less than ASSD. ASSD showed consistent accuracy on both pristine and synthetic speech. The size of the input vector shown in Table 3 indicates that ASSD uses much less data as input (30kb). In comparison, the time domain signal used by TSSDNet corresponds to ~1.5Mb; the spectrogram used by CCT correspods to ~1Mb.

### 4.4. Ablation Study

We conducted an ablation study to determine the choice of hyperparameters, training configuration, and the information from the compressed bit stream that leads to the best detection performance. In all experiments in this ablation study except the last (*i.e.*, Information from the compressed bit stream), we use all AAC bit stream information described in Table 1. In the ablation study, we use $D_{dev}$ to compute the performance. The ablation study results are shown in Table 4.

- Balanced sampling: we extracted all information from the compressed bit stream shown in Table 1 for the first 30 blocks/frames. For simplicity, we set $bshape$ and $bstride$ (Sec. 3) to 1. We found that the balanced sampling improved the balanced detection accuracy.
- Number of blocks: we fixed the balanced sampling and varied the number of blocks $L$ (Sec. 3). If the speech signal had less than $L$ blocks, we zero-padded the extracted information from the compressed bit stream. We found that $L = 30$ that correspond to 3.84 seconds, which is close to the average duration ( 3.3 seconds) of speech samples in ASVspoof2019 is the best choice.
- Loss: we fixed the number of blocks $L$ to 30 and varied the loss used for training. We compared the performance of ASSD trained with BCE loss and with BCE loss plus center loss (Sec. 3). The latter showed improved balanced accuracy.
- Choice of $bshape$: we used BCE loss plus center loss for training, fixed $bstride$ to 1, and varied $bshape$. We found that $bshape = 4$ performed the best. Note $bshape = 1$ means no block grouping.
- Information from the compressed bit stream: we examined the parameters extracted from the AAC bit stream (Table 1) and evaluated their impact on detection performance. We always used the temporal location of the block/frame (*i.e.*, `frame_indx`). Depending on the Windowing Scheme (WS) of each block, the number of scale factors can be 41 for `long` or 61 for `short` [18]. Since `short` windowing scheme is less frequently selected by the AAC encoder, we devised two strategies for processing the scale factors: 1) SF(41): only use the first 41 scale factors for both windowing schemes; 2) SF(61): use all 61 scale factors for `short` and pad the scale factors for `long` with 20 zeros. We found that the temporal block location (*i.e.*, `frame_indx`) and the scale factors (*i.e.*, `GlobalGain`, `pScaleFactor`) are the most important information from the compressed domain. In Experiment #2 we used only these two types of parameters from the AAC bit streams. Note that these parameters can be represented by 1kb per block.

**Table 4**. The ablation study results for different combinations of hyperparameters and configurations, as discussed in Sec. 4.4.

| Hyperparameters/ Configuration | | Balanced Accuracy (%) | AUPRC (%) |
|---|---|---|---|
| **Sampling** | w/o Balanced Sampling | 75.82 | 99.47 |
| | w/ Balanced Sampling | **77.89** | **99.60** |
| $L$ | 20 | 76.56 | 99.55 |
| | 25 | 75.94 | 99.52 |
| | 30 | **77.89** | **99.60** |
| | 35 | 73.83 | 99.52 |
| **Loss** | $\mathcal{L}_{BCE}$ | 77.89 | 99.60 |
| | $\mathcal{L}_{BCE} + \mathcal{L}_{center}$ | **79.13** | **99.63** |
| $bshape$ | 1 | 79.13 | 99.63 |
| | 2 | 82.54 | 99.69 |
| | 4 | **82.98** | **99.67** |
| | 8 | 80.93 | 99.65 |
| **Information from Compressed Bit stream** | All | 82.98 | 99.67 |
| | WS+MDCT+scale factors (SF)(41) | 83.67 | **99.72** |
| | WS | 50.00 | 90.90 |
| | MDCT | 53.46 | 92.35 |
| | SF(41) | 82.07 | 99.70 |
| | SF(61) | **84.10** | 99.70 |

## 5. CONCLUSION AND FUTURE WORK

We investigated the detection of AAC compressed synthetic speech. Our proposed method, ASSD, uses information from the compressed domain and does not require the decompressed speech signal. The experimental study shows that ASSD performs well when compared to other synthetic speech detection methods, given that it uses typically only 1kb per speech block from the compressed bit stream. This is much lower than other reported methods.

Future work will be devoted to two main tasks: 1) fusing between the proposed method and other techniques exploiting time-domain or spectrogram representations; 2) adapting our work to other compression schemes, for example MP3.

## 6. REFERENCES


[1] C. Borrelli, P. Bestagini, F. Antonacci, *et al.*, "Synthetic Speech Detection Through Short-term and Long-term Prediction Traces," *EURASIP Journal on Information Security*, vol. 2021, no. 1, p. 2, Apr. 2021.

[2] D. H. Klatt, "Review of Text-to-Speech Conversion for English," *The Journal of the Acoustical Society of America*, vol. 82, no. 3, pp. 737–793, May 1987.

[3] J. Kim, J. Kong, and J. Son, "Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech," *Proceedings of the International Conference on Machine Learning*, vol. 139, pp. 5530–5540, Jul. 2021, Virtual.

[4] T. Wang, R. Fu, J. Yi, *et al.*, "Prosody and Voice Factorization for Few-Shot Speaker Adaptation in the Challenge M2voc 2021," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8603–8607, Jun. 2021, Toronto, Canada.

[5] V. Popov, I. Vovk, V. Gogoryan, *et al.*, "Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech," *Proceedings of the International Conference on Machine Learning*, vol. 139, pp. 8599–8608, Jul. 2021, Virtual.

[6] K. Zhou, B. Sisman, R. Liu, and H. Li, "Seen and Unseen Emotional Style Transfer for Voice Conversion with A New Emotional Speech Dataset," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 920–924, Jun. 2021, Toronto, Canada.

[7] Y. Ren, C. Hu, X. Tan, *et al.*, "FastSpeech 2: Fast and High-Quality End-to-End Text to Speech," *Proceedings of the International Conference on Learning Representations*, pp. 1–15, May 2021, virtual.

[8] Y. Gao, R. Singh, and B. Raj, "Voice Impersonation Using Generative Adversarial Networks," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2506–2510, Apr. 2018, Calgary, Canada.

[9] S. Arik, J. Chen, K. Peng, *et al.*, "Neural Voice Cloning with a Few Samples," *Proceedings of the Advances in Neural Information Processing Systems*, vol. 31, pp. 10 019–10 029, Dec. 2018, Montreal, Canada.

[10] X. Tian, S. W. Lee, Z. Wu, *et al.*, "An Exemplar-Based Approach to Frequency Warping for Voice Conversion," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1863–1876, Jul. 2017.

[11] B. Smith, "Goldman Sachs, Ozy Media and a $40 Million Conference Call Gone Wrong," *The New York Times*, Sep. 2021. [Online]. Available: https://www.nytimes.com/2021/09/26/business/media/ozy-media-goldman-sachs.html.

[12] F. Akdeniz and Y. Becerikli, "Detection of Copy-Move Forgery in Audio Signal with Mel Frequency and Delta-Mel Frequency Kepstrum Coefficients," *Proceedings of the Innovations in Intelligent Systems and Applications Conference*, pp. 1–6, Oct. 2021, Elazig, Turkey.

[13] F. Hassan and A. Javed, "Voice Spoofing Countermeasure for Synthetic Speech Detection," *Proceedings of the International Conference on Artificial Intelligence*, pp. 209–212, Apr. 2021, Settat, Morocco.

[14] K. Bhagtani, A. K. S. Yadav, E. R. Bartusiak, *et al.*, "An Overview of Recent Work in Media Forensics: Methods and Threats," *arXiv preprint arXiv:2204.12067*, Apr. 2022.

[15] E. R. Bartusiak and E. J. Delp, "Synthesized Speech Detection Using Convolutional Transformer-Based Spectrogram Analysis," *Proceedings of the IEEE Asilomar Conference on Signals, Systems, and Computers*, pp. 1426–1430, Oct. 2021, Asilomar, CA.

[16] 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, Jun. 2021.

[17] International Organization for Standardization/International Electrotechnical Commission, *ISO/IEC 13818-7:1997 Information technology - Generic Coding of Moving Pictures and Associated Audio Information - Part 7: Advanced Audio Coding (AAC)*, 1997. [Online]. Available: https://www.iso.org/standard/25040.html.

[18] J. Herre and H. Purnhagen, "General Audio Coding," in *The MPEG-4 Book*, F. C. Pereira and T. Ebrahimi, Eds., Upper Saddle River, NJ, USA: Prentice Hall PTR, 2002, pp. 487–544.

[19] X. Li, N. Li, C. Weng, *et al.*, "Replay and Synthetic Speech Detection with Res2Net Architecture," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6354–6358, Jun. 2021, Toronto, Canada.

[20] M. Sahidullah and G. Saha, "Design, Analysis, and Experimental Evaluation of Block Based Transformation in MFCC Computation for Speaker Recognition," *Speech Communication*, vol. 54, pp. 543–565, 4 May 2012.

[21] B. Bogert, M. Healy, and J. Tukey, "The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking," *Proceedings of the Symposium on Time Series Analysis*, vol. 15, pp. 209–243, Jun. 1963, New York, NY.

[22] T. B. Patel and H. A. Patil, "Cochlear Filter and Instantaneous Frequency Based Features for Spoofed Speech Detection," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 618–631, Dec. 2017.

[23] M. Todisco, H. Delgado, and N. Evans, "Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification," *Computer Speech & Language*, vol. 45, pp. 516–535, Sep. 2017.

[24] M. Zakariah, M. K. Khan, and H. Malik, "Digital Multimedia Audio Forensics: Past, Present and Future," *Multimedia Tools and Applications*, vol. 77, no. 1, pp. 1009–1040, Jan. 2017.

[25] J. Khochare, C. Joshi, B. Yenarkar, *et al.*, "A Deep Learning Framework for Audio Deepfake Detection," *Arabian Journal for Science and Engineering*, vol. 47, pp. 3447–3458, 3 Nov. 2021.

[26] L. Rabiner and R. Schafer, *Theory and Applications of Digital Speech Processing*, 1st. USA: Prentice Hall Press, 2010.

[27] S. Stevens, J. Volkmann, and E. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *Journal of the Acoustical Society of America*, vol. 8, pp. 185–190, 3 Jun. 1937.

[28] P. Ma, R. Wang, D. Yan, and C. Jin, "A Huffman Table Index Based Approach to Detect Double MP3 Compression," *Digital-Forensics and Watermarking*, vol. 8389, pp. 258–271, Jan. 2014.

[29] Z. Xiang, P. Bestagini, S. Tubaro, and E. J. Delp, "Forensic Analysis and Localization of Multiply Compressed MP3 Audio Using Transformers," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 2929–2933, 2022, Singapore.

[30] International Organization for Standardization/International Electrotechnical Commission, *ISO/IEC 13818-3:1995 Information technology - Generic Coding of Moving Pictures and Associated Audio Information - Part 3: Audio*, 1995. [Online]. Available: https://www.iso.org/standard/22991.html.

[31] Q. Huang, R. Wang, D. Yan, and J. Zhang, "AAC Double Compression Audio Detection Algorithm based on the Difference of Scale Factor," *Information*, vol. 9, no. 7, p. 161, 2018.

[32] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is All You Need," *Proceedings of the Neural Information Processing Systems*, Dec. 2017, Long Beach, CA.

[33] A. K. S. Yadav, E. Bartusiak, K. Bhagtani, and E. J. Delp, "Synthetic Speech Attribution using Self Supervised Audio Spectrogram Transformer," *Proceedings of the IS&T Media Watermarking, Security, and Forensics Conference, Electronic Imaging Symposium*, Jan. 2023.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *Proceedings of the International Conference on Learning Representations*, May 2021, Virtual.

[35] Y. Gong, C.-I. Lai, Y.-A. Chung, and J. Glass, "SSAST: Self-Supervised Audio Spectrogram Transformer," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, pp. 10 699–10 709, Oct. 2022, Virtual.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, Jun. 2019, Seattle, Washington.

[37] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," *Proceedings of the ISCA Interspeech*, pp. 571–575, Aug. 2021, Brno, Czech Republic.

[38] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient Training of Audio Transformers with Patchout," *arXiv preprint arXiv:2110.05069*, Oct. 2021.

[39] K. Brandenburg, "MP3 and AAC Explained," *Proceedings of the 17th International Conference of Audio Engineering Society: High-Quality Audio Coding*, 1999, Signa, Italy.

[40] T. Painter and A. Spanias, "Perceptual Coding of Digital Audio," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451–515, 2000.

[41] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A Discriminative Feature Learning Approach for Deep Face Recognition," *Proceedings of the European Conference on Computer Vision*, pp. 499–515, Oct. 2016, Amsterdam, Netherlands.

[42] M. Todisco, X. Wang, V. Vestman, *et al.*, "ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection," *Proceedings of the Annual Conference of the International Speech Communication Association*, Sep. 2019, Graz, Austria.

[43] J. Yamagishi, M. Todisco, M. Sahidullah, *et al.*, *ASVspoof 2019: The 3rd Automatic Speaker Verification Spoofing and Countermeasures Challenge database*, Mar. 2019. [Online]. Available: https://www.asvspoof.org/index2019.html.

[44] Google Inc., *YouTube Recommended Upload Encoding Settings*, 2022. [Online]. Available: https://support.google.com/youtube/answer/1722171.

[45] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *Proceedings of the International Conference for Learning Representations*, May 2015, San Diego, CA.

[46] A. Tharwat, "Classification Assessment Methods," *Applied Computing and Informatics*, vol. 17, no. 1, pp. 168–192, Dec. 2021.

[47] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The Balanced Accuracy and Its Posterior Distribution," *Proceedings of the 2010 20th International Conference on Pattern Recognition*, pp. 3121–3124, 2010, Istanbul, Turkey.

[48] K. Boyd, K. H. Eng, and C. D. Page, "Area Under the Precision-Recall Curve: Point Estimates and Confidence Intervals," *Proceedings of 2013 Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 451–466, 2013, Prague, Czech Republic.