MULTIPLE-POINT INPUT AND TIME-INVERTED SPEECH SIGNAL FOR THE ASVspoof 2021 CHALLENGE

Sunghyun Yoon¹ and Ha-Jin Yu²

¹ Department of Artificial Intelligence, Kongju National University, Republic of Korea
² School of Computer Science, University of Seoul, Republic of Korea

ysh901108@naver.com, hjyu@uos.ac.kr

ABSTRACT
This paper describes the replay attack detection system we submitted to the ASVspoof 2021 challenge. We used two approaches to build the system: 1) multiple-point input for convolutional neural networks (CNNs) and 2) using the phase spectrum of the time-inverted speech signal. Using multiple-point input improves detection accuracy by increasing the amount of information available at a single time. Combining the phase spectra of the original signal and the time-inverted signal reduces the within-class variance, resulting in higher detection accuracy. For each method, we built several subsystems using four kinds of CNN-based networks and five kinds of the combination methods. The final system we submitted was obtained by averaging the scores of all the subsystems. We achieved a min t-DCF of 0.7648 and an EER of 29.55% in the evaluation trials of ASVspoof 2021 physical access scenario.

1. INTRODUCTION
Automatic speaker verification (ASV) is one of the promising biometrics owing to its stable verification accuracy and convenience of usage. However, ASV is vulnerable to spoofing attacks. Spoofing attacks attempt to deceive ASV systems by using a spoofed utterance produced to sound like another specific speaker’s utterance. The spoofed utterance is produced using methods such as text-to-speech synthesis, voice conversion, and replay attack. ASV systems cannot distinguish between genuine and spoofed utterances, and this vulnerability cannot be mitigated by merely improving the verification accuracy; specific countermeasures against spoofing attacks are needed.

With this in mind, spoofing detection for ASV has been the focus of considerable research in recent years. Spoofing detection is the task of distinguishing between genuine and spoofed utterances and thus protecting ASV from spoofing attacks. It is based on the fact that there are differences in frequency attributes between genuine and spoofed utterances [1, 2].

ASV spoofing and countermeasures (ASVspoof) challenges are being periodically held for research and development on spoofing detection techniques [3, 4, 5]. ASVspoof 2021 [6] was the fourth such competition. Unlike previous challenges, which had focused on simulated environments, ASVspoof 2021 intended to develop robust spoofing detection systems in more practical and realistic environments. The training and development datasets of ASVspoof 2021 were identical to those of ASVspoof 2019, which were collected in simulated environments. However, the evaluation dataset of ASVspoof 2021 was designed under realistic environments, unlike in previous challenges. Therefore, the key issue of ASVspoof 2021 was how to develop systems that are robust in real environments using only the simulated corpus. ASVspoof 2021 comprised three tasks: logical access (LA), physical access (PA), and speech deepfake (DF). We competed only in the PA section, whose goal was to detect replay attacks.

Many state-of-the-art spoofing detection systems are based on convolutional neural networks (CNNs). The attributes that can distinguish between genuine and replayed utterances (e.g., the attributes generated by playback/recording devices) are spread across the entire time domain without time dependency. These attributes can be efficiently captured by CNN. In our experiments, we used four kinds of CNN-based networks, which will be introduced in Section 3.2.

In order to build the systems, we utilized two methods that we had proposed in our earlier papers. The first is the use of multiple-point input for CNNs [7], and the second is the use of a combination of the phase spectra of the original as well as time-inverted speech signals [8]. We confirmed in our previous studies that the former method reduces detection errors in replay attack detection by increasing the amount of available information at one time, which achieved a 44% relative improvements in the ASVspoof 2019 PA task. We also confirmed that the latter method reduces the intra-class variances by generating unseen intra-class conditions, which achieved 12% and 31% relative improvements in ASVspoof 2019 LA and PA tasks, respectively. Therefore, we expect that we could achieve considerable performance improvement over the baseline systems in the ASVspoof 2021 PA task, by combining these two proposed methods.

The remainder of this paper is organized as follows. Section 2 describes the methods used to build our spoofing detection systems. Section 3 describes the experimental setup. In Section 4, the obtained results are analyzed. Finally, Section 5 concludes the paper.

2. SYSTEM DESCRIPTION
In this section, we describe two approaches used in building our replay attack detection system. One is the use of multiple-point input for CNN, while the other is using the phase spectra of the original as well as time-inverted speech signals.

* Corresponding author: Ha-Jin Yu (hjyu@uos.ac.kr)
2.1. Multiple-point input for CNN

The use of multiple-point input for CNN was devised to overcome the challenges of handling variable-length features (e.g., acoustic features from speech) when using CNNs.

Like other kinds of neural networks, CNNs are also trained in mini-batch unit. To feed input features into a CNN in mini-batch units, all the features in each mini-batch must have the same length. However, spoken utterances usually have varying lengths, and these lengths are not known in advance. Therefore, we need to ensure that all the features in each mini-batch have the same length before feeding them into the CNN.

Feature segmentation [9, 10] is an efficient method for handling variable-length input features. It breaks a variable-length feature $X$ that has a length of $T$ into fixed-length segments $\{F_i\}_{i=0}^{N-1}$ using a sliding window having a length of $M(< T)$ frames and is shifted by $L$-frames interval at each step. For the case $M \geq T$, the feature $X$ is padded to make the length $M$. The padded feature becomes a segment $F_0$.

After the segmentation, CNN takes one segment $F_i$ at one time, rather than an entire feature $X$. In the training phase, each segment is considered an individual input, and the class label of a feature is shared across all the segments from the feature. In the evaluation phase, the score of a sample is computed by averaging the scores of all the segments from that sample.

With the feature segmentation, however, CNN can consider only limited amount of information in a segment concurrently, not the entire information in a feature. To address this lacuna, we proposed the use of multiple-point input for CNN [7] based on bidirectional features segmentation. Figure 1 provides an overview of bidirectional feature segmentation. Unlike conventional feature segmentation, where a feature is broken down along a single direction, bidirectional feature segmentation breaks a feature along two directions: positive and negative time directions. Therefore, two sets of segments are obtained from one feature: one is a set of forward segments $\{F_i\}_{i=0}^{N-1}$ (obtained along the positive time direction) and the other is a set of backward segments $\{B_i\}_{i=0}^{N-1}$ (obtained along the negative time direction). When $M \geq T$, we can get one forward segment $F_0$ and one backward segment $B_0$, by padding $X$ (i.e., the original feature) and $\overline{X}$ (i.e., the time-inverted frame order of $X$), respectively. From these two sets, we take the $i$-th forward and backward segments, $F_i$ and $B_i$, to form a pair.

The CNN takes a pair of two segments as input together. We can double the amount of information available at one time because the two segments in each pair generally cover different time ranges. There are several kinds of combination methods that use both segments together simultaneously, which will be discussed in Section 2.3.

2.2. Phase spectrum of time-inverted speech signal

This method is based on phase spectrum. Unlike magnitude spectrum, phase spectrum has the time-inverting property: when the time order of a signal is inverted, the values of its phase spectrum are changed. This means that the attributes of the phase spectrum are changed when the time order is inverted. From this fact, we assumed that when the time order is inverted, the identities related to intra-class variations (e.g., phrase, speaker, and language information) are changed, but those related to inter-class variations (e.g., the information of playback and recording devices) are not.

Motivated by this assumption, we proposed the method of using the phase spectra of the original speech signal $x(n)$ and time-inverted speech signal $\tilde{x}(n)$ together [8]. Figure 2 illustrates the framework of the proposed method. With the proposed method, two kinds of phase spectra, $\tau$ and $\tilde{\tau}$, are obtained from one speech signal. $\tau$ is the phase spectrum of the original signal $x(n)$, and $\tilde{\tau}$ is the phase spectrum of the time-inverted signal $\tilde{x}(n)$.

The CNN takes these phase spectra, $\tau$ and $\tilde{\tau}$, as input concurrently, similar to the method of multiple-point input. However, the time ranges of both phase spectra are the same in
this method. The same combination methods that are used in
the method of multiple-point input can be used for the proposed
method to use both phase spectra together concurrently.

2.3. Combination methods

Both methods introduced in Sections 2.1 and 2.2 have the CNN
taking two input features concurrently. In order for the CNN
to utilize the information of these two features simultaneously, we
proposed five kinds of combination methods in [8], which are
categorized into three classes, as illustrated in Figure 3.

Figure 3b shows the framework of the two-channel input method
called 2ch, where two features are concatenated at the input
channel-level. Note that all the acoustic features we used in
our experiments are single-channel, which has the shape (1 × 
T × D). After concatenating the two features, we can obtain a
two-channel feature, which has the shape (2 × T × D). The
CNN takes this two-channel feature as input. Therefore, the
number of parameters for the first layer of the CNN doubles.

Figure 3c shows the framework of the embedding-level combination,
where the embedding stands for the output of the
Figure 3d shows the framework of the feature map-level
combination, where the feature map stands for the output of the
last layer of the CNN. No additional parameters are required
because the two feature maps are obtained using the shared
CNN. The feature maps are then combined by taking the
element-wise maximum (called fmax) to form a single feature
map that is used to compute an embedding. We considered
neither concatenation along the channel axis nor element-wise
averaging, because both produce the same outputs as concat
and vmean, respectively.

3. EXPERIMENTS

3.1. Database

All the experiments in this study were conducted upon the
ASVspoof 2021 PA database, where the training and
development sets are the same as the ASVspoof 2019 PA
database, but the evaluation set differs. The training set contains
5,400 genuine and 48,600 spoofed utterances. The development
set contains 5,400 genuine and 24,300 spoofed utterances. The
evaluation set contains 943,110 utterances, but the class of each
utterance is unknown. We used the training set for building the
systems, and used the development set for validating the
detection errors for every epoch. We did not perform any other
kinds of data augmentation, except the time-inversion.

3.2. Experimental setup

As input features, we used a 257-dimensional log power
spectrogram for the multiple-point input, and a 257-
With both methods, we first extracted 25-ms frames from a
signal at intervals of 10 ms. The frame-level pre-processing
was performed as follows: removing DC offset, pre-emphasis with
a coefficient of 0.97, and Hamming windowing. The number of
FFT points was 512. For the feature segmentation, used in both
proposed methods, we set the segment length $M = 600$ and the
shift interval $L = 300$.

We used four kinds of CNN-based networks: SE-ResNet-
34 [12, 13], DenseNet-121 [14], ShuffleNetV2 with 0.5x output
channels [15], and MNASNet with depth multiplier of 1.0 [16].
All the networks have a softmax classifier with two classes (i.e.,
genuine and spoof). The score was defined as the log ratio of
the spoofed probability to the genuine probability, $\log p_g(x) −
\log p_s(x)$. All the weights were initialized from the He_normal
distribution [17], and no bias was used.

There are two kinds of proposed methods, four kinds of
CNN-based networks, and five kinds of combination methods
in our experiments. By combining them, we built $40(= 2 \times 4\
5)$ kinds of spoofing detection systems. All systems were
trained for 100 epochs with a mini-batch size of 64. We used
AMSGrad [18] to train each system, a variant of the Adam [19]
onmizer with a learning rate of $10^{-3}$. $\beta_1 = 0.9$, $\beta_2 = 0.999$,
$\epsilon = 10^{-8}$, and a weight decay of $10^{-4}$. We implemented the
systems using PyTorch [20].

For each system, we selected up to two models. One is the
best model based on min t-DCF [21] and the other is the best
model based on equal error rate (EER). Only one model was
selected if the best model for min t-DCF is equal to that for
EER. We then fused all the systems at the score-level by simply
averaging. We did not perform score normalization.

}\end{center}

\textbf{Figure 3.} Frameworks of the (a) conventional and (b-d)
proposed combination methods. The proposed methods consist
of (b) two-channel input, (c) embedding-level concatenation,
and (d) feature map-level combination.
4. RESULTS

Table 1 presents the min t-DCF and EER of the systems on the development trials. Each system in Table 1 (i.e., corresponding to a row, except the last row) stands for the fusion of the subsystems at the score-level, where each subsystem was built using the same proposed method and network structure, but various kinds of the combination methods. For example, the system listed in the first row of Table 1 (i.e., built with the method of multiple-point input and the SE-ResNet) was obtained by fusing five corresponding subsystems, that is, multiple-point input-based SE-ResNet built with concat, vmin, vmean, fmax, and 2ch, respectively. The system in the last row (denoted as Fusion) stands for the fusion of all the subsystems listed in Table 1 at the score-level.

From the results in Table 1, we observed that the multiple-point input method showed lower detection errors than the time-inverted speech signal method. In addition, the fused system (row 9) showed higher detection errors than any of the multiple-point input method-based systems (rows 1–4) that are based on the magnitude spectrum. This is mainly because the magnitude spectrum is generally more efficient for replay attack detection than the phase spectrum. Even so, the phase spectrum contains useful information for replay attack detection that the magnitude spectrum does not contain [22]. Furthermore, we expected the method of time-inverted speech signal to show robust performances in unseen environments, because this method has the effect of generating the speeches of unseen intra-class conditions [8]. For these reasons, we used magnitude as well as phase spectra to design the final system.

Table 1. Min t-DCF and EER (%) of the systems on the development trials (progress phase)

<table>
<thead>
<tr>
<th>Method</th>
<th>Network</th>
<th>Min t-DCF</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple-point input</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE-ResNet</td>
<td>0.0060</td>
<td>0.1677</td>
<td></td>
</tr>
<tr>
<td>DenseNet</td>
<td>0.0039</td>
<td>0.1286</td>
<td></td>
</tr>
<tr>
<td>ShuffleNet</td>
<td>0.0055</td>
<td>0.2067</td>
<td></td>
</tr>
<tr>
<td>MNASNet</td>
<td>0.0030</td>
<td>0.0936</td>
<td></td>
</tr>
<tr>
<td>Phase spectrum of time-inverted speech signal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE-ResNet</td>
<td>0.0274</td>
<td>0.9259</td>
<td></td>
</tr>
<tr>
<td>DenseNet</td>
<td>0.0467</td>
<td>1.4444</td>
<td></td>
</tr>
<tr>
<td>ShuffleNet</td>
<td>0.0508</td>
<td>1.5535</td>
<td></td>
</tr>
<tr>
<td>MNASNet</td>
<td>0.0265</td>
<td>0.7778</td>
<td></td>
</tr>
<tr>
<td>Fusion</td>
<td>0.0152</td>
<td>0.5010</td>
<td></td>
</tr>
</tbody>
</table>

Table 2 presents the min t-DCF and EER of the baseline submitted systems on the evaluation trials, where the final system we submitted to ASVspoof 2021 was obtained using score-level fusion of all the subsystems listed in Table 1. Our system significantly outperformed the baselines of the ASVspoof 2021 PA task. The min t-DCFs and EERs were relatively reduced on average by approximately 21.7% (18.9% to 23.5%) and 30.2% (22.4% to 39.2%), respectively.

However, there are large gaps in performance between the development and evaluation trials. The min t-DCF and EER on the development trials (see Fusion in Table 1) increased by a large extent in the evaluation trials, from 0.0152 and 0.5010% to 0.7648 and 29.55%, respectively. When the systems based on the method of time-inverted speech signal (rows 5–8 in Table 1) are not used, those on the evaluation trials increase even more, from 0.7648 and 29.55% to 0.9997 and 46.03% respectively. Considering the fact that the training and development sets were collected under simulated conditions but the evaluation set was collected under more realistic conditions, we can conclude that it is still difficult to build robust systems for a real environment using only simulated utterances.

Table 2. Min t-DCF and EER (%) of the baseline and final submitted systems on the evaluation trials (evaluation phase)

<table>
<thead>
<tr>
<th>System</th>
<th>Min t-DCF</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQCC-GMM</td>
<td>0.9434</td>
<td>38.07</td>
</tr>
<tr>
<td>LFCC-GMM</td>
<td>0.9724</td>
<td>39.54</td>
</tr>
<tr>
<td>LFCC-LCNN</td>
<td>0.9958</td>
<td>44.77</td>
</tr>
<tr>
<td>RawNet2</td>
<td>0.9997</td>
<td>48.60</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.7648</td>
<td>29.55</td>
</tr>
</tbody>
</table>

5. CONCLUSION

In this study, we built replay attack detection systems for the ASVspoof 2021 PA task. The systems were built based on the approaches of multiple-point input and time-inverted speech signal, both of which we had proposed in our earlier papers. The proposed methods improve the detection accuracy by increasing the amount of information available at a single time and reducing the intra-class variances. For each of the proposed methods, we built several systems using various CNN-based networks and the combination methods. Then, we computed the final score by averaging the scores of all the systems. The submitted system achieved considerable improvement over the ASVspoof 2021 PA baselines.

However, the proposed method still showed substantial detection errors, making it unsuitable for deployment in practical applications. Further research is required to bridge the gap of the detection accuracies between simulated and real environments. In the future, therefore, we will investigate how to build robust spoofing detection systems in realistic conditions using only simulated datasets.

6. ACKNOWLEDGMENT

This research was supported and funded by the Korean National Police Agency. [Project Name: Real-time speaker recognition via voiceprint analysis / Project Number: PR01-02-040-17].
7. REFERENCES


