Learning Efficient Representations for Fake Speech Detection

Nishant Subramani, Delip Rao*
AI Foundation
San Francisco, California
{nishant, delip}@aifoundation.com

Abstract

Synthetic speech or "fake speech" which matches personal vocal traits has become better and cheaper due to advances in deep learning-based speech synthesis and voice conversion approaches. This increased accessibility of synthetic speech systems and the growing misuse of them highlights the critical need to build countermeasures. Furthermore, new synthesis models evolve all the time and the efficacy of previously trained detection models on these unseen attack vectors is poor. In this paper, we focus on: 1) How can we build highly **accurate, yet parameter and sample-efficient** models for fake speech detection? 2) How can we **rapidly adapt detection models** to new sources of fake speech? We present four parameter-efficient convolutional architectures for fake speech detection with best detection F1 scores of around 97 points on a large dataset of fake and bonafide speech. We show how the fake speech detection task naturally lends itself to a novel multi-task problem further improving F1 scores for a mere 0.5% increase in model parameters. Our multi-task setting also helps in data-sparse situations, commonplace in adversarial settings. We investigate an alternative approach to the data-sparsity problem using transfer learning and show that it is possible to meet purely supervised detection performance for unseen attack vectors with as little as 6.25% of the training data. This is the first known application of transfer learning in adversarial settings for speech. Finally, we show how well our transfer learning approach adapts in an instance-efficient way to new attack vectors using the Real-Time Voice Cloning toolkit. We exceed the purely supervised detection performance (99.18 F1) with as little as 6.25% of the data.

Introduction

Our ability to synthesize human voice with the likeness of a subject, referred to as "synthetic audio" or "fake speech," has dramatically improved in the last 3 years with progress in speech synthesis (SS), voice conversion (VC), and general deep learning technology. Neural text to speech models such as WaveNet (Oord et al., 2016), Tacotron (Wang et al., 2017), and DeepVoice (Gibiansky et al., 2017; Ping et al., 2017) have been extended to incorporate speaker identity Arik et al. (2018); Nachmani et al. (2018). Other latent attributes such as prosody have also been successfully used for conditioning the speech synthesis — Wang et al. (2018); Skerry-Ryan et al. (2018); Akuzawa, Iwasawa, and Matsuo (2018) — resulting in realistic personalized speech that is becoming harder for humans to distinguish. Such high-quality and personalized synthetic speech has many positive applications such as building natural conversational assistants[1], creating personalized answering machines, and restoring the voice of people with speech pathologies, to mention a few.

However, this technology has also been used for malicious purposes such as spreading misinformation and disinformation, using a person's voice without consent[2], creating falsified material (evidence tampering), and committing various cybercrimes including harassment and intimidation. In multiple recent incidents, perpetrators used synthetic speech technology to impersonate CEOs and other leaders to commit fraudulent wire-transfer thefts[3]. According to a 2018 survey, the rate of voice fraud increased over 350 percent from 2013 to 2017. It is estimated that voice fraud costs U.S. organizations $14 billion each year within call centers alone[4]. Improvements in speech synthesis and voice conversion research, and the availability of easy to use toolkits, such as Project VoCo by Jin et al. (2017), which calls itself the "photoshop of voice" and the Real-Time Voice Cloning Toolkit[5], which promises to "clone a voice in 5 seconds to generate arbitrary speech in real-time," only exacerbate the problems with synthetic voices.

In this work, we examine synthetic voice from an adversarial lens, where each synthesis model is viewed as an "attack vector." Our goal is to build highly accurate, parameter efficient fake speech detection models that can be deployed

*Corresponding Author

[1] See Google Duplex: https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html

[2] See this example where Prof. Jordan Peterson's voice is cloned without consent: https://www.vice.com/en_us/article/43kwgb/not-jordan-peterson-voice-generator-shut-down-deepfakes

[3] See for example: https://www.wsj.com/articles/fraudsters-use-ai-to-mimic-ceos-voice-in-unusual-cybercrime-case-11567157402

[4] https://www.securitymagazine.com/articles/89432-voice-fraud-climbs-350

[5] https://github.com/CorentinJ/Real-Time-Voice-Cloning

widely across devices of different memory and compute capabilities. In this setting, the threat landscape continuously shifts with the daily emergence of new synthesis models. As a result, we also want to react quickly to this "shifting adversary" by adapting existing pretrained detection models to new attack vectors (synthesis models). Our key contributions in this paper are:

- We present two lightweight convolutional network models (EfficientCNN and RES-EfficientCNN) for synthetic speech detection that achieve over 97 macro-F1 with fewer than 50,000 parameters and less than a 100 KB memory footprint. Further, this model takes less than a third of the MFLOPS that resnet18 takes.
- For a marginal increase in number of parameters, we present a novel multi-task setting that improves detection performance without using any new labeling information (source of the attack vector is implicit supervision). We believe this is the first paper to jointly model the veracity (bonafide vs. fake) and the source of the fakes.
- We present the first-ever study of transfer learning in adversarial settings for speech and apply it to the growing problem of audio fakes. In resource-poor conditions that are typical of adversarial situations (adapting to new attack vectors), we demonstrate how transfer learning not only drastically cuts down the amount of training data needed, but also, in many cases, exceeds purely supervised performance. Further, we show the robustness of our transfer learning approach under a variety of settings:
  1. neural-to-neural speech synthesis
  2. neural-to-non-neural voice conversion
  3. neural-to-non-neural speech synthesis to unit selection
  4. neural-to-neural speech synthesis to voice conversion

  Across the board we see that our transfer learning approach is able to match fully-supervised performance with as little as 6.25% of training data.
- We present a new attack vector, the Real Time Voice Cloning Toolkit, and show how existing fake speech detection models can be fine-tuned to surpass purely supervised detection performance (99.12 F1) with as little as 6.25% of the data.

Datasets

We present results on two sets of fake speech datasets, one from the ASVSpoof2019 challenge and another from a new dataset we introduce in this paper called RTVCSpoof.

ASVSpoof2019

We use datasets from Todisco et al. (2019), originally created for the ASVSpoof 2019 challenge. We only use the logical access (LA) portion of the dataset, i.e. we do not use any knowledge of the channel and ambient parameters of the recording. The data has two sets of labels – one binary (BONAFIDE vs. FAKE) and a "source" label which can take 1 of 6 values for spoofed audio (FAKE). The source labels include, SS1, SS2, SS4 for three different neural network-based speech synthesis systems, US1 for a unit-selection based speech synthesis system, and VC1, VC4 for a neural-network based and a transform-function based voice conversion system respectively. We exploit both sets of labels for our multi-task setup as described later. Table 1 summarizes the data used in our experiments. The ratio of male and female speakers in the training and validation sets are identical (40% male and 60% female). The length of the audio samples in the training and validation sets are also near identical, with a minimum, median, and maximum lengths around 650, 3300, and 12000 milliseconds respectively.

Table 1: Summary of ASVSpoof2019 data used here

	Training	**Validation**
#speakers (disjoint)	20	10
#samples	25,380	5,438
Dataset prior, $P(\text{FAKE})$	.898	.897

For our models, we use the full training set, and report results on the validation set. We did not have access to the official ASVSpoof2019 test set.

RTVCSpoof

We develop a speaker-specific synthetic speech synthesis (5S) recipe to generate this dataset. Our recipe uses a set of utterances of a speaker, a set of text sequences, and a pretrained speech synthesis model to generate synthetic waveforms for each text sequence in the speaker's voice.

As our speech synthesis model, we choose the Real Time Voice Cloning Toolkit (RTVC) (Jemine, 2019). Note that this recipe can use any speech synthesis model. RTVC allows users to clone voices with as little as a single 5-second sample. To clone a voice using this model, we feed a short reference utterance of the target speaker into a speaker encoder, which results in a speaker embedding used to condition a synthesizer. We take the mean of all speaker utterances for an individual speaker as our speaker embedding. Then, we input a sequence of text processed as a sequence of phonemes as well as the speaker embedding into the synthesizer, resulting in a log mel spectrogram. To generate the spoofed speech waveform, we pass the spectrogram into a

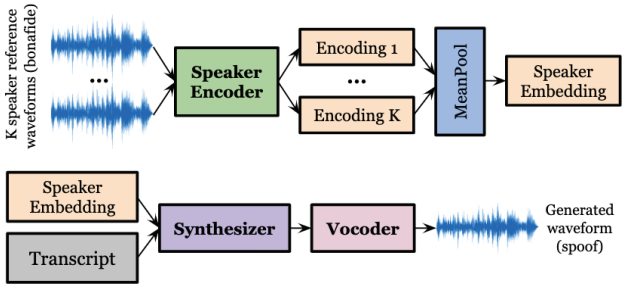

Figure 1: 5S recipe to generate RTVCSpoof: we take K speaker waveforms as input and compute a speaker embedding. This speaker embedding is used to generate a synthetic waveform that corresponds to an input transcript. Here boxes in bold refer to components of the pretrained RTVC model.

vocoder. Since any text sequence can be input into our 5S recipe and the RTVC model, we can quickly generate as many utterances for a single speaker as needed. Refer to Figure 1 for details.

Using our 5S recipe with RTVC as our synthesis model, we construct a synthetic dataset called RTVCSpoof. RTVCSpoof has 3284 bonafide utterances and 4843 spoofed utterances from 369 distinct speakers of various genders. These speakers have rich demographic information varying in age from teens to eighties and varying in accents ranging from Indian to English to African to Malaysian among others.

Choosing Bonafide Examples

We select 226 speakers from the Mozilla Common Voice project on the English side which have 3 or more contributed utterances and have demographic information[6]. For each speaker with 3 to 19 utterances, we select all of their utterances. For those with more than 19 utterances, we randomly sample 19 utterances to include in our dataset. These become our BONAFIDE examples in our training set.

For the validation set, we select 143 distinct speakers with the same qualification from the preset validation portion of the dataset. Each speaker has 3 to 11 contributed utterances. All of these become the BONAFIDE examples in our validation set.

Generating Spoofed Examples

We generate each FAKE example using a speaker utterance as a template. For a target speaker, we take all of their utterances that were chosen above and compute speaker embeddings for each. Next, we compute the mean speaker embedding over all these selected utterances. We condition the RTVC pretrained synthesizer with this mean speaker embedding. To construct the training set, we randomly choose 17 sentences from the English side of the IWSLT16[7] English to German translation training set. We make sure that the sentence lengths for the chosen sentences are approximately between 10 and 25 words. To construct the validation set, we follow the same procedure as above, but select 7 sentences at random from the validation set of the English side of the IWSLT16 English to German translation dataset. See Figure 1 for more details.

Demographic Information & Dataset Splits

Our 5S recipe produces a set of 2624 BONAFIDE and 3842 FAKE utterances for training as well as 660 BONAFIDE and 1001 FAKE examples for validation. We chose these sizes to mimic the class distribution for an attack vector in ASVSpoof2019 and ensure that no speakers overlap between the training (226 speakers) and validation (143 speakers) sets. Our speakers belong to a wide range of demographics summarized in Table 2 for training and Table 3 for validation.

[6]The Common Voice Project has a CC0 license, allowing us to do this

[7]https://workshop2016.iwslt.org/

Table 2: Demographic information (age, gender, and accent) for RTVCSpoof (training).

Age	#	Gender	#	Accent	#
Twenties	91	Male	199	US	131
Thirties	58	Female	26	England	36
Fourties	34	———	—	Indian	12
Others	43	Others	1	Others	47

Table 3: Demographic information (age, gender, and accent) for RTVCSpoof (validation).

Age	#	Gender	#	Accent	#
Twenties	58	Male	130	US	67
Thirties	42	Female	12	England	21
Fourties	16	———	—	Indian	25
Others	27	Others	1	Others	30

Preprocessing and Feature Extraction

We truncate long audio samples to 4 seconds by selecting the first 4 seconds of audio. For audio shorter than 4 seconds, we loop the samples to 4 seconds length. This length normalization makes sure all samples in a batch are of the same length. The four second limit is simply the ceiling of the median of all audio lengths in the training set.

For feature extraction, we convert each audio file into a spectrogram using a sampling rate of 16 GHz, 1728 FFTs, and a hamming window with a length of 108ms and a 10ms window shift to return a spectrogram of short-term Fourier transform magnitudes similar to Chettri et al. (2018). Lastly, we take the log of the spectrogram values and Z-normalize to generate our input representation for each audio file. All our experiments use this standard input representation. We leave exploring other input representations to future work.

Models

In this paper, we study four different convolutional models for fake speech detection: EfficientCNN, its residual form, RES-EfficientCNN, and the multi-task variants of those two.

EfficientCNN

Our convolutional network model for synthetic speech detection is a variant of the model used by Wu et al. (2018), which we call EfficientCNN. The input to the EfficientCNN model is the normalized log-spectrogram as described in the previous section, which is processed through an input processing block and four convolution blocks. The output of the pooling layer after the final convolution block is passed to a classification block for obtaining the classifier predictions. We now describe each of these blocks in detail; see Figure 2 for a summary.

Input Processing Block The input processing block takes the normalized log-spectrogram input and passes it through a 2d convolution layer with a 5x5 kernel, a ReLU activation layer, a batch normalization layer, and a max-pooling layer.

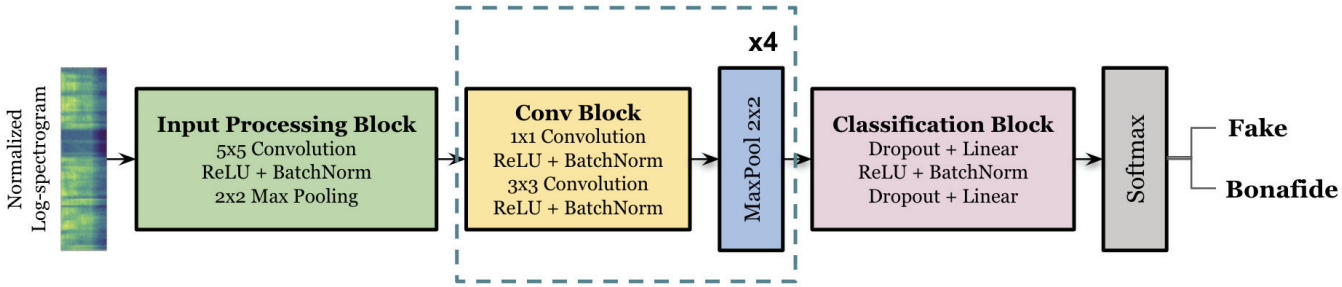

Figure 2: EfficientCNN – A model consisting of an input processing block, 4 convolution blocks, and a classification block.

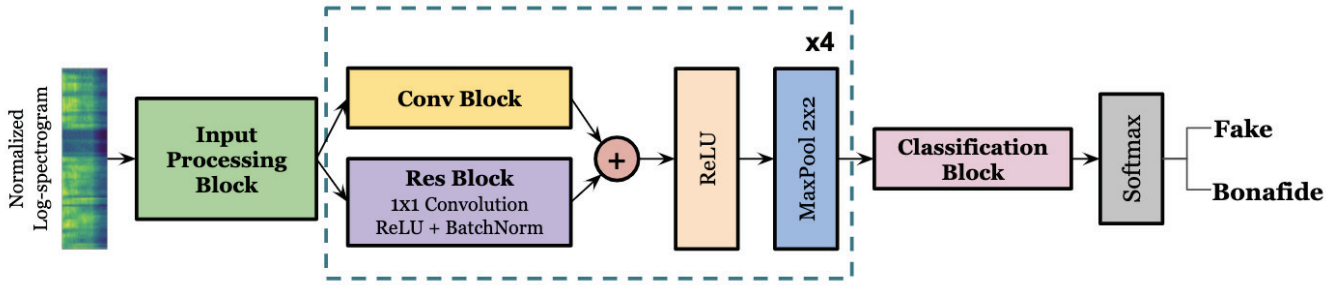

Figure 3: The RES-EfficientCNN is a residual version of the EfficientCNN model described in Figure 1.

Convolution Block Each convolution block consists of a 1x1 convolution, ReLU, batch normalization, a 3x3 convolution, another ReLU, and another batch normalization. The output of this block is passed through a 2x2 max-pooling layer before becoming the input to another convolutional or classification block.

Classification Block The classification block consists of a linear layer, a ReLU, a batch normalization layer, and another linear layer with dropout included before each linear layer. The output of the final linear layer is passed to a softmax layer to produce model predictions.

RES-EfficientCNN

We also experiment with a residual version of EfficientCNN, that we call RES-EfficientCNN. Residual connections in convolutional networks have been used with great success in computer vision – see He et al. (2016) – to improve training stability and accuracy. The RES-EfficientCNN is similar to EfficientCNN, in that they have identical input processing, convolution, and classification blocks, but with one difference: each convolution block has corresponding residual block, and the output of a convolution block is added to the output of its corresponding residual block. This combined output becomes the input to the next convolution block as illustrated in Figure 3.

Residual Block Typical residual connections simply combine the input x with the output of a layer $\mathcal{F}(x)$ as $x+\mathcal{F}(x)$. We designed the residual block to do this in spirit, but we massage the input x to match the dimension of $\mathcal{F}(x)$ by performing a 1x1-convolution – see Lin, Chen, and Yan (2013) – followed by a ReLU and batchnorm.

Multi-Task Variants

Finally, we introduce multi-task variants of EfficientCNN and RES-EfficientCNN, called Multi-EfficientCNN and Multi-RES-EfficientCNN respectively, by noting that any good learned convolution representation should not only be able to predict if the input audio was BONAFIDE or FAKE, but also predict the source where it came from. From the ASVSpoof 2019 data, we have 7 possible values for source: SS1, SS2, SS4, US1, VC1, VC4, and NONE, where NONE labels bonafide speech. Please refer to the Datasets section for additional details about these sources. To incorporate these two tasks, we modify the EfficientCNN and RES-EfficientCNN architectures to include two classification blocks. The cross entropy losses from each classification block are summed up and this joint loss is minimized during training via backpropagation. During inference, we discard the source prediction head and the corresponding classification block and use the rest of the model exactly like the single task setup. We illustrate this in Figure 4.

Hyperparameters and Training Details

To train our models, we use mini-batch stochastic gradient descent (SGD) to minimize weighted cross-entropy loss or the cumulative weighted cross-entropy loss in case of the multi-task variants. These weights are computed by taking the inverse class abundance, i.e., if the class distribution is 99% and 1%, the class with 1% will have a weight of 99 and

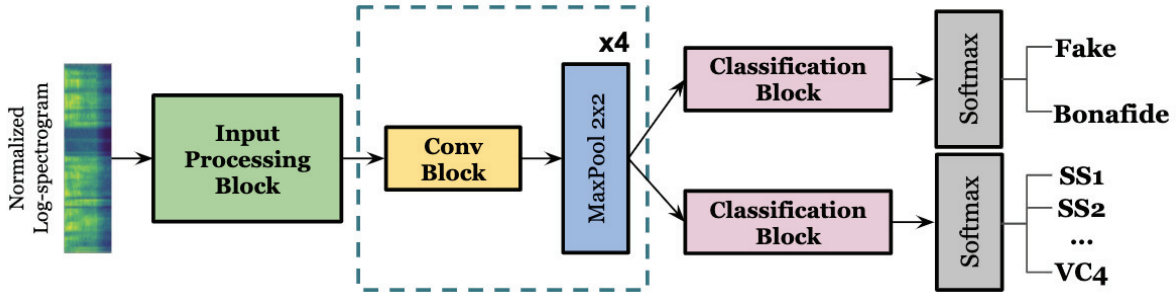

Figure 4: Multi-task training setup: we have two classification blocks, one for each task – fake speech detection and speech source detection. The fake speech detection task is our task of interest (FAKE vs. BONAFIDE), while the Speech Source detection task classifies the attack vector (synthesis/voice conversion model used for fake speech or NONE for bonafide speech).

the class with 99% will have a weight of 1. This handles the massive class imbalance we observe in the data. We choose the model with the lowest weighted cross entropy loss during training and use that for evaluation on held-out data. For our experiments, we use a single NVIDIA V100 GPU unless otherwise specified.

Model Hyperparameters Our 5x5 convolutional layer has a stride length of 2 and padding size of 2. All of our 3x3 convolutional layers, network-in-network 1x1 layers, and residual 1x1 convolutional layers have a stride length of 1 and no padding. All of our max-pooling layers use a stride length of 2. We initialize all weight matrices using Xavier normal initialization and use dropout of 0.2 on the input to the linear layers of both our models. For optimization, we use mini-batch SGD with a batch size of 128 and an Adam optimizer with default parameters ($\alpha = 10^{-3}, \beta = [0.9, 0.999]$) (Kingma and Ba, 2014). We halve the learning rate, α, during training whenever there is no improvement in validation set loss: completely stopping when the learning rate drops below 10^{-5}. Note: we did not excessively tune or optimize hyperparameters, so there could be better performing hyperparameter configurations.

Experiments and Results

We experiment with EfficientCNN and RES-EfficientCNN models of different capacities – SMALL, MEDIUM, LARGE. The differences between these variants is summarized in Table 4. We vary the capacity of the models by setting the number of filters in the input processing block to 2, 4, and 8 for SMALL, MEDIUM, and LARGE respectively. For the largest model (LARGE) each successive convolutional block has number of input filters equal to 12, 16, 12, and 8. This scales down by a factor of 2 and 4 for the MEDIUM (6, 8, 6, and 4) and SMALL (3, 4, 3, and 2) models respectively. Note that our small model takes only 9 MFLOPs and largest model takes just 256 MFLOPs, which are 100x and 3x less compute respectively than resnet18.

In our experiments, as summarized in Table 5, we find that: 1) increasing number of paramaters helps with this dataset, 2) the residual versions of the models, RES-EfficientCNN and Multi-RES-EfficientCNN, consistently perform better than their non-residual counterparts and, 3) the multi-task learning setup performs better than the single task setup for the SMALL and MEDIUM models, where macro F1 gains of around 7% and 11% are observed for a fixed cost of adding 231 parameters in training for the multi-task setup. These results are significant with standard deviations of less than 1.0 F1 points for all models considered. Further, when doing difference of means hypothesis tests between Multi-RES-EfficientCNN and Multi-EfficientCNN as well as RES-EfficientCNN and EfficientCNN on the large setting, we find p-values less than 0.00001 in both cases, indicating that the residual variants are significantly better.

Table 4: Number of parameters for the three variants of the four different models we experiment. Notice the multi-task versions have fixed parameter cost of 231 additional parameters over the single-task ones.

	SMALL	MEDIUM	LARGE
EfficientCNN	7654	15690	34114
+RES	10580	21794	46610
+Multi	7885	15921	34345
+Multi & RES	10811	22025	46841

During inference, there is no additional parameter cost incurred because of multi-task learning as the classification block unrelated to the FAKE vs. BONAFIDE prediction can be discarded. This has important consequences in improving accuracies of low footprint detection models on mobile devices. Our best performing model, multi-task variant of RES-EfficientCNN, achieves a macro F1 score of 97.61 on the validation set.

Multi-task Learning and Data Sparsity

In adversarial settings, it is common to not have sufficient data to train models from scratch. To study the efficacy of our multi-task training setup, we performed dataset ablation experiments with our best performing multi-task model, Multi-RES-EfficientCNN and compared with its non-multi-task variant, RES-EfficientCNN. The results are summarized in Figure 5.

Again, we notice that the multi-task setup consistently

Table 5: Results (Macro-F1 on validation set) of the four different model variants under three different parameterizations. As observed elsewhere, the residual versions perform better than the non-residual ones, and the multi-task variants perform better than the non-multi-task ones.

	SMALL	MEDIUM	LARGE
EfficientCNN	88.44	85.36	94.14
+RES	86.91	86.05	97.22
+Multi	88.44	93.41	86.00
+Multi & RES	**92.91**	**95.91**	**97.61**

Figure 5: Comparison between Multi-RES-EfficientCNN and RES-EfficientCNN models for varying amounts of data: for small data sizes, the multi-task setup produces better results than the single-task setup.

performed significantly better than the single-task setup. This is remarkable because for a nominal increase in training cost, by adding the 231 parameters needed for the multi-task setup, we see a consistent gain of 10 F1 points for situations with less data.

Adversarial Detection Performance

We consider 4 different adversarial situations where the models are exposed to attack vectors (synthesis / voice conversion models) they have not observed before in training. For each situation, we report the macro F1 scores on the new attack vector.

- **Condition A: Neural to Neural Synthesis**. In this condition, we assume the pretrained model in deployment was trained on data from neural speech synthesis models and was exposed to an unseen and different neural speech synthesis model. To study this condition, we train our best RES-EfficientCNN model on the SS1 and SS2 portion of the data, and evaluate it on the SS4 portion of the data.
- **Condition B: Neural to Non-Neural Voice Conversion**. Here we assume the model deployed in production is trained on data from a neural voice conversion system and evaluated on data from a non-neural voice conversion system. To study this, we train our best RES-EfficientCNN model on the VC1 portion of the data, and evaluate it on the VC4 portion of the data.
- **Condition C: Neural to Unit Selection Synthesis**. Here we assume the model deployed in production is trained on data from neural speech synthesis systems and the attack vector is from a unit selection speech synthesis system. For this condition, we train our best RES-EfficientCNN model on the SS1, SS2, and SS4 portions of the data, and evaluate it on the US1 portion of the data.
- **Condition D: Neural Synthesis to Neural Voice Conversion**. In this condition, we assume the pretrained model in deployment was trained on data from neural speech synthesis models and were exposed to data from a neural voice conversion system. To study this condition, we train our best RES-EfficientCNN model on the SS1, SS2, and SS4 portions of the data, and evaluate it on the VC1 portion of the data.

These four conditions and their results are summarized in Figure 6. It is clear that models deployed in production perform poorly not only when the attack vector belongs to a different class (speech synthesis vs. voice conversion), but also within the same class (different kinds of speech synthesis / voice conversion). Adversarial conditions B and D have especially low adversarial detection performance. The poor performance in condition B stems from two facts: (1) generalization from a neural to a non-neural voice conversion system is difficult and (2) the VC4 attack vector is more difficult to detect than others even with many examples of the attack vector as seen in Figure 6. We attribute the poor adversarial detection performance for adversarial condition D to the difference in distributions between the speech synthesis systems and voice conversion systems. In the next section, we show how transfer learning can be used effectively to deal with these adversarial attacks with very little data.

Adapting to New Adversaries with Transfer Learning

We wanted to study how well transfer learning techniques are suited for handling changing adversaries. We consider the same four adversarial conditions (**A**, **B**, **C**, and **D**) detailed in the previous section, except we assume we have access to a small portion of the data corresponding to the new attack vector (target) to fine-tune the classifier in production. We perform dataset ablation experiments where we increase the availability of the target data ranging from 6.25% to 100%, doubling at each interval. The results for the four conditions are summarized in Figure 6. The dotted blue line in the charts refers to a purely supervised macro F1 score as evaluated on the validation set, i.e. trained on 100% of the target data without any pretrained model initialization. Two important observations arise:

- In all situations only a small portion of target data (between 6.25% to 25%) is needed to meet or exceed supervised performance.
- We can exceed purely supervised performance for many situations by training with a pretrained initialization, demonstrating the value of transfer learning in building

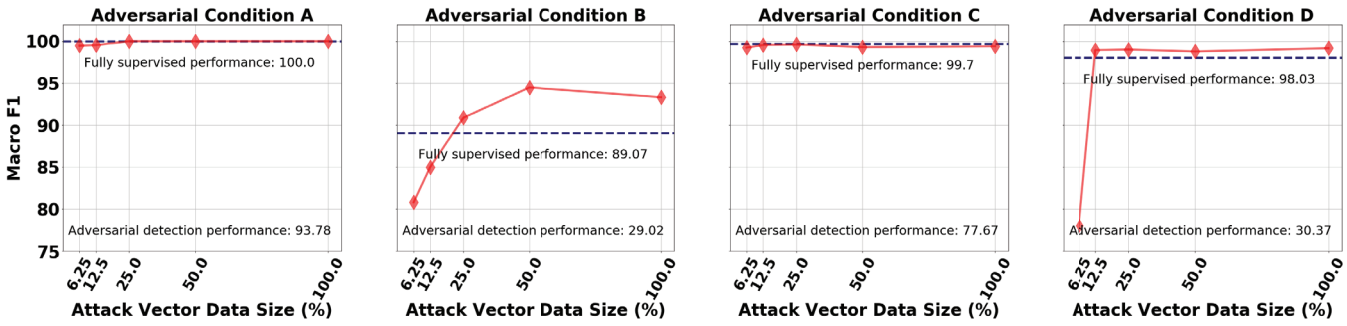

Figure 6: Adapting to new adversaries using transfer learning: The dotted red line indicates purely supervised performance. Note that we are able to meet or exceed the purely supervised performance for less than 25% of the training data with transfer learning.

detection models even in situations where data sparsity is not an issue. Training by fine-tuning a pretrained model seems to be a general good practice for building any detection model.

Results on RTVCSpoof Dataset

We repeat our experiments on the new RTVCSpoof dataset with the largest RES-EfficientCNN model. For this experiment, we treat the RTVCSpoof dataset as an unseen attack vector and use a model trained on the neural speech synthesis portion of ASVSpoof (i.e. the SS1, SS2, and SS4 portions) dataset. With this setup, we achieve an adversarial detection performance (i.e. without fine-tuning) of 91.35 macro-F1 on the validation portion of RTVCSpoof dataset. With transfer learning, we are able to achieve near purely supervised performance with around 6.25% of the data. Figure 7 summarizes the results.

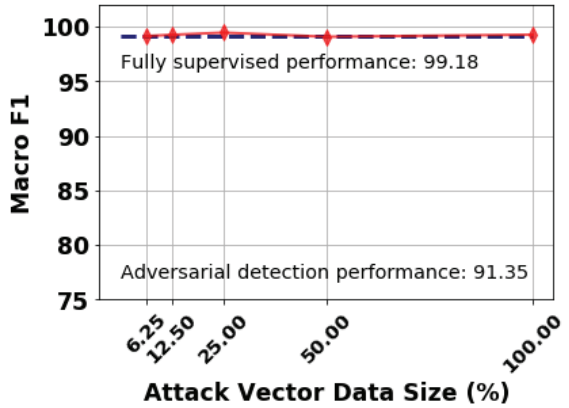

Figure 7: Results for adapting to RTVCSpoof dataset. We are able to achieve 99.12 F1 with as little as 6.25% of the data using transfer learning.

Related Work

Audio Classification: Convolutional networks have become popular for audio classification tasks starting from Piczak (2015) and Salamon and Bello (2017). Our work focuses on a parameter efficient CNN model inspired by Wu et al. (2018) that we call EfficientCNN. Motivated by the massive improvements in computer vision by residual architectures – first proposed by He et al. (2016) – we further propose a residual version of EfficientCNN, called RES-EfficientCNN, that improves detection significantly.

Speaker Verification: The first applications of synthetic speech detection was in speaker verification, where the goal is to identify a bonafide utterance from a speaker vs. utterances derived via synthesis models[8]. For our experiments, we use the ASVSpoof2019 dataset and focus on the situation where the room and ambient parameters are unknown.

Multimedia Forensics: Convolutional network models were successfully used by Rössler et al. (2018) to detect manipulated images. Most audio forensics work comes from the speaker verification spoofing community as discussed earlier. Given the wide ranging impact of the adversarial uses of speech synthesis, broader studies outside of the speaker verification context is warranted. That is the goal of our current work.

Multi-task Learning: This paper also presents a previously unexplored multi-task learning setup, where the learner is jointly optimized on multiple related tasks, to improve detection accuracies. We refer the readers to Zhang and Yang (2017) for a general review of multi-task learning.

Transfer Learning: Transfer learning has been used successfully for a variety of tasks in computer vision and natural language processing (Pan and Yang, 2009; Yosinski et al., 2014). In Cozzolino et al. (2018) the authors use transfer learning for image forensics. To our knowledge, this paper is the first to show efficacy of transfer learning techniques to detecting synthetic/fake speech from novel sources.

Conclusion

In an effort to combat the rising abuse of synthetic speech systems, we have presented parameter efficient models for fake speech detection and demonstrated how to make them instance-efficient using transfer learning. Our models, EfficientCNN and RES-EfficientCNN, achieve better than 97

[8] asvspoof.org

macro F1 points on a large fake speech dataset. These models need fewer than 50,000 parameters and have around 100 KB memory footprint. We proposed a novel multi-task learning formulation for the fake speech detection task, which improves detection performance without needing to use any additional data or excessive computational resources. We also showed the multi-task setup enables better generalization for our leaner versions of the EfficientCNN and RES-EfficientCNN models.

Since new attack vectors (speech synthesis and voice conversion systems) are being developed to fool state-of-the-art detectors, adaptation these new attack vectors is critical. We presented a transfer-learning based approach to adapt to unseen attack vectors and evaluated our transfer-learning in four different adversarial settings: neural-to-neural speech synthesis, neural-to-non-neural voice conversion, neural-to-non-neural speech synthesis to unit selection, and neural-to-neural speech synthesis to voice conversion. Our method matches purely-supervised performance with as little as 6.25% of the training data in most of these settings. Finally, we showed how the transfer learning approach is broadly-applicable by considering a new attack vector, the Real Time Voice Cloning Toolkit and fine-tuned an existing detection model to it. With this, we were able to surpass the purely supervised performance (99.12 F1) using as little as 240 examples (around 6% of the entire dataset).

Acknowledgements

We would like to thank Intel Corporation for their gracious donation of Google Cloud Credits that made some of the experimentation possible and Gaurav Bharaj, Jesse Berman, Jonathan Chu, and Eric Yang for helpful comments on versions of this draft.

References

Akuzawa, K.; Iwasawa, Y.; and Matsuo, Y. 2018. Expressive speech synthesis via modeling expressions with variational autoencoder. *arXiv preprint arXiv:1804.02135*.

Arik, S.; Chen, J.; Peng, K.; Ping, W.; and Zhou, Y. 2018. Neural voice cloning with a few samples. In *Advances in Neural Information Processing Systems*, 10019–10029.

Chettri, B.; Mishra, S.; Sturm, B. L.; and Benetos, E. 2018. Analysing the predictions of a cnn-based replay spoofing detection system. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, 92–97. IEEE.

Cozzolino, D.; Thies, J.; Rössler, A.; Riess, C.; Nießner, M.; and Verdoliva, L. 2018. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*.

Gibiansky, A.; Arik, S.; Diamos, G.; Miller, J.; Peng, K.; Ping, W.; Raiman, J.; and Zhou, Y. 2017. Deep voice 2: Multi-speaker neural text-to-speech. In *Advances in neural information processing systems*, 2962–2970.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jemine, C. 2019. Master thesis: Automatic multispeaker voice cloning.

Jin, Z.; Mysore, G. J.; Diverdi, S.; Lu, J.; and Finkelstein, A. 2017. Voco: Text-based insertion and replacement in audio narration. *ACM Transactions on Graphics (TOG)* 36(4):96.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lin, M.; Chen, Q.; and Yan, S. 2013. Network in network. *arXiv preprint arXiv:1312.4400*.

Nachmani, E.; Polyak, A.; Taigman, Y.; and Wolf, L. 2018. Fitting new speakers based on a short untranscribed sample. *arXiv preprint arXiv:1802.06984*.

Oord, A. v. d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Pan, S. J., and Yang, Q. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22(10):1345–1359.

Piczak, K. J. 2015. Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 1–6. IEEE.

Ping, W.; Peng, K.; Gibiansky, A.; Arik, S. O.; Kannan, A.; Narang, S.; Raiman, J.; and Miller, J. 2017. Deep voice 3: Scaling text-to-speech with convolutional sequence learning. *arXiv preprint arXiv:1710.07654*.

Rössler, A.; Cozzolino, D.; Verdoliva, L.; Riess, C.; Thies, J.; and Nießner, M. 2018. Faceforensics: A large-scale video dataset for forgery detection in human faces. *arXiv preprint arXiv:1803.09179*.

Salamon, J., and Bello, J. P. 2017. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters* 24(3):279–283.

Skerry-Ryan, R.; Battenberg, E.; Xiao, Y.; Wang, Y.; Stanton, D.; Shor, J.; Weiss, R. J.; Clark, R.; and Saurous, R. A. 2018. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. *arXiv preprint arXiv:1803.09047*.

Todisco, M.; Wang, X.; Vestman, V.; Sahidullah, M.; Delgado, H.; Nautsch, A.; Yamagishi, J.; Evans, N.; Kinnunen, T.; and Lee, K. A. 2019. Asvspoof 2019: Future horizons in spoofed and fake audio detection. *arXiv preprint arXiv:1904.05441*.

Wang, Y.; Skerry-Ryan, R.; Stanton, D.; Wu, Y.; Weiss, R. J.; Jaitly, N.; Yang, Z.; Xiao, Y.; Chen, Z.; Bengio, S.; et al. 2017. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*.

Wang, Y.; Stanton, D.; Zhang, Y.; Skerry-Ryan, R.; Battenberg, E.; Shor, J.; Xiao, Y.; Ren, F.; Jia, Y.; and Saurous, R. A. 2018. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. *arXiv preprint arXiv:1803.09017*.

Wu, X.; He, R.; Sun, Z.; and Tan, T. 2018. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security* 13(11):2884–2896.

Yosinski, J.; Clune, J.; Bengio, Y.; and Lipson, H. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, 3320–3328.

Zhang, Y., and Yang, Q. 2017. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*.