INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Robust Bayesian and Light Neural Networks for Voice Spoofing Detection

Radosław Białobrzeski [*], *Michał Kośmider* [*], *Mateusz Matuszewski* [*], *Marcin Plata* [*], *Alexander Rakowski* [*]

Samsung R&D Institute Poland

{r.bialobrzes, m.kosmider, m.matuszews2, m.plata, a.rakowski2}@samsung.com

Abstract

We present a replay attack detection system consisting of two convolutional neural network models. The first model consists of a small Bayesian neural network, motivated by the hypothesis that Bayesian models are robust to overfitting. The second one uses a bigger architecture, LCNN, extended with several regularization techniques to improve generalization. Our experiments, considering both size of the networks and use of the Bayesian approach, indicated that smaller networks are sufficient to achieve competitive results. To better estimate the performance against unseen spoofing methods, the final models were selected using novel Attack-Out Cross-Validation. In this procedure each model was tested on a subset of data containing not only previously unseen speakers, but also unseen spoofing attacks. The system was submitted to ASVspoof 2019 challenge's PA condition and achieved a t-DCF score of 0.0219 and EER of 0.88% on the evaluation dataset, which is a 10 times relative improvement over the baseline.

Index Terms: anti-spoofing, automatic speaker verification, speaker recognition, spoofing countermeasures, deep learning, bayesian neural networks

1. Introduction

Speaker recognition is becoming one of the preferred authentication methods in various scenarios, especially in environments where voice control is the device interaction method of choice, e.g. Smart Home devices such as smart speakers. As with most biometric characteristics, speaker recognition is vulnerable to spoofing [1, 2] and a presentation attack detector is needed to secure the system from any spoof attempts such as replay, voice conversion or voice synthesis. This need is especially important nowadays, when the more sophisticated attack methods are already difficult to detect even by a human operator [3]. To help researchers compare their work using common baselines, datasets and test protocols, ASVspoof challenge was first organized in 2015 and has continued in a bi-yearly fashion.

In the latest edition, ASVspoof 2019, the dataset consisted of two distinct data conditions: physical access (PA) and logical access (LA). The former focused on attacks where the spoofer replays genuine voice samples to the device (e.g. a smart speaker). The latter mimicked a situation where the spoofer can submit arbitrary voice samples directly to a certain service (which may be remote), making use of techniques such as voice conversion. Results were to be evaluated using not only the equal error rate (EER) metric, but also with the tandem decision cost function (t-DCF) [4], to account for the joint performance (or conversely, error) of both the speaker verification and anti-spoofing systems.

This work describes the Samsung R&D Institute Poland submission to the PA condition and the observations made during the submitted systems' design and development. Our approach to tackling limited training data and model generalization resulted in a simple yet effective primary system, achieving 6th position in the overall ranking of primary systems.

*All authors contributed equally to this work.

2. Related Work

Different approaches of improving replay attack countermeasures have been investigated in recent years. New features, system architectures and training procedure modifications all contributed to a major leap in state-of-the-art spoof detection quality. On the feature level, perhaps the most important development is the observation that traditional features like mel-frequency cepstral coefficients may not be optimal in this setting, as most of the discriminative information is contained in the higher frequencies [5, 6]. Different novel features were proposed, most notably constant Q cepstral coefficients, but their applications are limited due to domain mismatch vulnerability and poor generalization [7]. Pursuit of the optimal system architecture resulted in multiple novel neural classifiers, consistently surpassing traditional GMM and i-vector counterparts [6, 8]. The training procedures of these classifiers can be improved by applying techniques such as data augmentation [8] or multi-task learning [9].

3. Models

3.1. Bayesian NN

The primary model is a Bayesian neural network [10]. Using a Bayesian approach is partially motivated by a relatively small training dataset. Bayesian neural networks are believed to be less prone to overfitting [11, 12], therefore this task seems like an appropriate application for them. The main goal of the challenge was generalization to unseen forms of spoofing [13]. Our primary interest was in the ability to capture the uncertainty that these types of networks possess. Scores predicted by traditional neural networks can be overly confident even for examples far away from the training data [14, 12]. This may be of some significance for selecting the decision threshold.

The model utilizes *Flipout* [15] modifications of convolutional and fully connected layers. Architecture details are shown in Table 1. A Bayesian neural network with weights w outputs probability $P(\hat{y} \mid x, w)$ of a sample x being classified as spoofed $\hat{y}$. A standard normal distribution prior $P(w)$ is imposed on each weight of the model. Posteriors $Q(w \mid \theta)$ are restricted to normal distributions parameterized by θ. Output of the model is the expectation of probabilities for multiple networks, approximated using empirical mean for n networks:

$$P(\hat{y} \mid x) \approx \frac{1}{n} \sum_{i=1}^{n} P(\hat{y} \mid x, w_i) \tag{1}$$

Here $P(\hat{y} \mid x, w)$ is the Bernoulli distribution parameterized by

http://dx.doi.org/10.21437/Interspeech.2019-2676

Table 1: *Model architecture of the Bayesian neural network.*

Layer Type	Parameters
Conv2DFlipout	8 filters, 5×5 kernel, padding, ReLU
MaxPooling2D	2×2 pooling, 2×2 stride, padding
BatchNorm	-
Conv2DFlipout	16 filters, 5×5 kernel, padding, ReLU
MaxPooling2D	2×2 pooling, 2×2 stride, padding
BatchNorm	-
Conv2DFlipout	48 filters, 5×5 kernel, padding, ReLU
MaxPooling2D	2×2 pooling, 2×2 stride, padding
GlobalMaxPool2D	-
BatchNorm	-
DenseFlipout	24 hidden units, ReLU
BatchNorm	-
DenseFlipout	1 hidden unit, Sigmoid activation

Table 2: *Architecture of the Light-CNN model.*

Layer Type	Parameters
Conv2D	32 filters, 5×5 kernel
BatchNorm	-
MaxFeatureMap	-
MaxPooling2D	2×2 pooling, 2×2 stride
*ConvBlock**	32, 48
*ConvBlock**	48, 64
*ConvBlock**	64, 32
*ConvBlock**	32, 32
Dense	64 hidden units
Dropout	0.7
MaxFeatureMap	
Dense	1 hidden unit, Sigmoid
**ConvBlock*(F_1, F_2)* structure:	
Conv2D	F_1 filters, 1×1 kernel
BatchNorm	-
MaxFeatureMap	-
Conv2D	F_2 filters, 3×3 kernel
BatchNorm	-
MaxFeatureMap	-
MaxPooling2D	2×2 pooling, 2×2 stride

a neural network. We find model parameters θ by minimizing Evidence Lower Bound [12, 10].

Input to the network is a logarithmically scaled mel-spectrogram of the entire audio file. Spectrograms have 512 mel-bins and are created using Short-time Fourier Transform with hop length of 512 and Hann window of size 2048 samples. Every spectrogram is repeated and truncated to match 280 columns.

3.2. Light-CNN (LCNN)

The second model utilized a Light-CNN (LCNN) model from [6] with an extended training procedure, involving regularization techniques and data augmentation. The main concept behind the LCNN architecture is the use of Max-Feature-Map activations instead of the commonly used ReLU function.

Similar to [6] input data were preprocessed by computing logarithms of power spectrograms using the Fast Fourier Transform. After standardizing the values to a *z-score*, spectrograms longer than 400 frames were truncated, while those shorter were padded with zeros. Mean and standard deviation used for standardization were computed across the whole training set. The resulting shape of the input data was equal to 864×400.

The network architecture follows that of [6] (which was in turn based on [16]) - see Table 2. Because this model could be more prone to overfitting compared to the Bayesian network, the following techniques were employed during training, to increase its robustness and generalization performance:

- **Multi-Task Learning**. In a similar spirit to [9] (which also used the LCNN architecture) the model was trained to predict additional metadata provided with the annotations, in addition to the original labels.
- **Manifold Mixup** [17]. This regularization technique was employed to prevent the model from yielding overly confident predictions for examples not seen in the training data. A robust anti-spoof model should not simply memorize a particular set of spoofing attacks. Predicting less confidently on data not similar to the training distribution should lower the risk of being exploited by a previously unencountered spoofing technique.
- **Augmentation**. The data augmentation procedure was inspired by the "mixing" technique from [18]. *Spoof* samples shorter than 10 seconds were randomly padded with cropped parts of *bonafide* samples, instead of zeros, with a probability of 50%. This should increase the difficulty of recognizing such samples, forcing the classifier to look for *spoofing* artifacts even in the presence of locally *genuine* parts.
- **Oversampling**. To account for class imbalance in the training set the probability of sampling a *bonafide* sample during training was increased 10 times.

4. Experiments

4.1. Dataset

The ASVspoof 2019 dataset for physical access (PA) was divided into three parts - *training*, *development* and *evaluation*, with the metadata specified only for the first two subsets. *Training* and *development* subsets contain *bonafide* and replayed speech of 40 speakers, disjointly divided into two sets of 20 speakers. Acoustic environment and replay attacks were simulated to ensure control over the acoustic and replay configurations. In order to generalize over unseen attack types with limited training data, we proposed and worked with another way of dividing the dataset which is described in more detail in Section 4.2.

For each sample, its acoustic environment was defined by three values - room size (square meters), T60 reverberation time (milliseconds) and distance from the talker to the ASV microphone (centimeters). All of these values were grouped into three bins. These were 2-5m^2, 5-10m^2 and 10-20m^2 for the room size, 50-200ms, 200-600ms and 600-1000ms for the T60 reverberation time, and 10-50cm 50-100cm and 100-150cm for the talker-to-ASV distance. An attack type was also distinguished by belonging to one of three levels of attacker-to-talker distance and one of three levels of replay device quality (*perfect*, *high*, *low*). The bins for the attacker-to-talker distance were 10-50cm, 50-100cm and above 100cm. These characteristics are marked with letters $\{A|B|C\}$ corresponding to respective bins in the same order as above and the first letter from the duple

Table 3: *Per-fold settings for the Attack-Out Cross-Validation setup. The duples $(A|B|C)^2$ denote attack identifiers from the ASVspoof 2019 Physical Access dataset. The first letter corresponds to "Attacker-to-talker distance" and the second to "Replay device quality" of an attack. These values are described in more detail in Subsection 4.1.*

Fold	Training Set	Validation Set	Test Set
0	BC, AA, CB, AB	BA, CA	BB, AC, CC
1	AB, CB, AC, BA	CC, BB	AA, BC, CA
2	CC, AA, CA, BB	BC, AC	CB, BA, AB

corresponds to the attacker-to-talker distance and the second to the replay device quality of an attack.

4.2. Attack-Out Cross-Validation

Overfitting to the training set is still an existing problem in state-of-the-art models for anti-spoofing. From the variety of models evaluated on the ASVspoof 2017 *Version 1* [2] and *2* [19] datasets only one exhibited a (slightly) better performance on the test data (see Table 2 in [20]). Due to the constant progress in machine learning-related fields new spoofing methods can emerge systematically. Therefore, it is important that the generalization performance is assessed not only on previously unseen speakers, but also, and probably more importantly, on unseen attacks.

Training and *development* datasets contained examples of only a portion of the possible spoofing methods, while the final *evaluation* dataset contained previously unseen ones. Thus, the performance measured on the *development* set might not reflect the final score of the models.

To obtain a better estimate of the true models' performance *Attack-Out Cross-Validation* was employed. In this setting, each model was trained and evaluated on 3 cross-validation folds, containing data from both the *training* and *development* sets. For each fold, 4 spoofing methods were used for training, 2 for validation and 3 for testing, so that each method appeared exactly once in a test set. Table 3 shows exact setups for each fold. Note that the *bonafide* samples were split in such a manner that a similar distribution over different speakers was maintained across the training/validation/test subsets.

This procedure was used to select the architectures and settings with the highest potential of generalization to unseen attacks. These - namely the Bayesian NN and LCNN models - were then trained on the training set only to obtain the final predictions.

4.3. Manual Data Analysis

Inspired by [21, 22] we conducted an analysis where we tried to define an embedding vector for particular types of attacks. Based on that we were also able to find similarities between particular attacks. For this purpose, we trained a neural network to learn the data embedding instead of a standard classification. The neural network contained three convolutional layers with 8, 16 and 32 filters, respectively, and four fully connected layers with 512, 256, 128 and 48 output features, respectively. The network returned the 48-element embedding vector normalized to unit length. We used the n-pair with angular loss function [23] which is a more efficient alternative to the standard triplet loss function. We tried to distinguish between all types of attacks and *bonafide*, i.e. we discriminated between various types of attacks during the training instead of binary classification (*genuine* vs. *spoof*). Afterwards, we applied t-SNE algorithm [24] for dimensionality reduction of the embedding vector to two main components.

The results of the data analysis were visualized in Figure 1. In the early stage of the training, the embedding vectors were strongly correlated with the replay device quality and *bonafide* samples were close to samples replayed with high quality devices. There was no visible fragmentation of the attacker-to-talker distance. After 50 epochs of training we still observed correlation with the replay device quality, but there was another visible split between the attacker-to-talker distance below $50cm$ and above $50cm$. At that point, *AA* attacks (attacks of the closest distance and the best quality) were still not separated from *bonafide* samples. After 200 epochs, we observed a separation of *bonafide* samples, but there was no separation of the attacker-to-talker distance above $50cm$ with the same replay device qualities. The analysis showed that the separation of the replay device quality parameters is less challenging. Despite a long training, the model did not reach complete separation of various attacker-to-talker distances. The problem of weak distinguishability of replay attacks with high quality devices and genuine samples seems to be critical for the PA task.

4.4. Bayesian NN

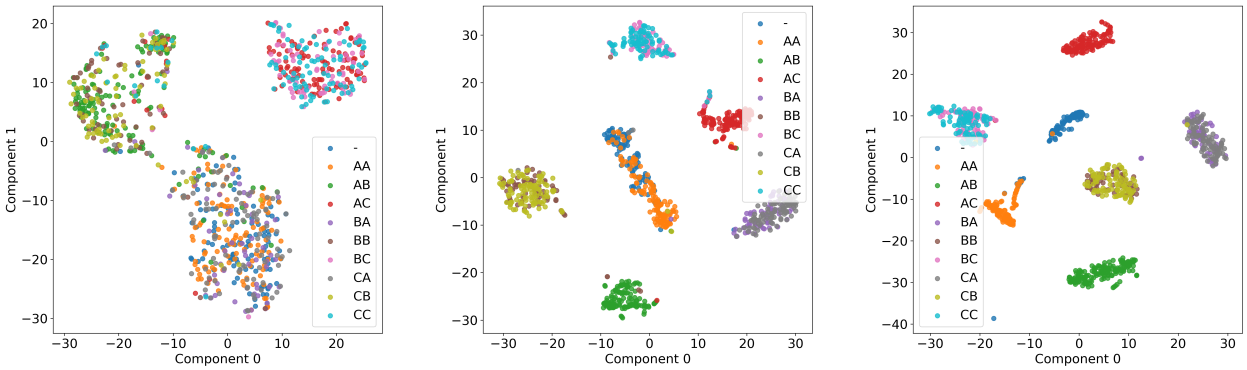
Bayesian NN model was trained using continuously oversampled dataset. Samples were drawn from each class equally often without disturbing the in class distributions. Optimization was performed using Adam [25] with learning rate of 0.001, β_1 equal 0.9 and β_2 equal 0.999. Mini-batch size was set to 64. Regularization term was divided by the size of the dataset.

4.5. LCNN

The LCNN model was trained for 20 epochs using the Adam optimizer [25] with a learning rate of 0.0001, on mini-batches of size 8. Manifold Mixup was used with α equal to 1. Additionally, each partial loss corresponding to the multi-task predictions (see Subsection 3.2) was weighted with 0.1.

4.6. Bayesian versus non-Bayesian NN

To validate our intuition taken after [12, 15], we conducted experiments on a model with Bayesian layers replaced with their standard counterparts. The experiment was conducted using the *Attack-Out Cross-Validation* method (see Subsection 4.2) on the ASVspoof 2019 PA dataset. The results on the validation subset showed that the model with standard layers, trained with the complete training set, outperformed the model with Bayesian layers. We also observed that in some cases the Bayesian neural network overfitted faster in comparison to the standard neural network which was an unexpected behavior. To validate whether this trend holds regardless of the amount of data, we repeated this experiment on a smaller version of the dataset, which contained only 20% of randomly chosen samples from the training dataset. In this case the Bayesian neural network proved its efficiency and generalized better compared to its "standard" counterpart. Results of the comparison are shown in Table 4. This might indicate that Bayesian NNs are indeed less prone to overfitting in scenarios where less data is available. However, it seems that the ASVspoof 2019 PA dataset is of size considerable enough for this effect to vanish.



(a) *After 3 epochs* (b) *After 50 epochs* (c) *After 200 epochs*

Figure 1: *A visualization of the data analysis results. The subfigures show embedding vectors after 3, 50 and 200 epochs of the embedding neural network training. The vectors was reduced to two main components with the t-SNE algorithm. The types of attacks are described in more detail in Subsection 4.1.*

Table 4: *Results of the comparison of Bayesian and non-Bayesian neural network models. The folds are described in Table 3.*

Type	**Full datastet**		**20% of the dataset**	
	min-tDCF	*ERR (%)*	*min-tDCF*	*ERR (%)*
		Fold 0		
Bayes	0.02561	0.00961	**0.04821**	**0.01536**
Non	**0.00912**	**0.00399**	0.05075	0.01667
		Fold 1		
Bayes	0.00014	7.36e−5	**0.00029**	**0.00014**
Non	**7.36e−5**	**3.68e−5**	0.00066	0.00042
		Fold 2		
Bayes	0.00072	0.00029	0.00315	0.0012
Non	**0.00018**	**9.09e−5**	**0.00261**	**0.00075**

4.7. Results

The final scores were obtained using a weighted average of predictions from a set of trained models. Per-model weights were obtained using a grid search procedure, to maximize results on the **cross-validation setup** (see Subsection 4.2). This was done by averaging test set results over all cross-validation folds, for a given set of weights. Note that while the weights for the ensembling were selected using the model parameters trained on the cross-validation setup, model parameters for the final predictions were optimized on the provided *training* dataset.

Table 5 shows results of the two models and their ensemble compared with the GMM baseline models. The ensemble of both models had a 10 times relative improvement over the baseline models on the two metrics measured, on both the development and evaluation datasets. These combined results also show an over 40% relative improvement over the Bayesian network, suggesting that features learned by the two models are complementary.

Table 5: *Results on the Development and Evaluation datasets. BNN used empirical mean from 128 samples. Ensemble weights were 59% and 41% for BNN and LCNN respectively.*

Dataset	*Development*		*Evaluation*	
Model	*min-tDCF*	*ERR (%)*	*min-tDCF*	*ERR (%)*
		Baseline models:		
LFCC-GMM	0.2554	11.96	0.3017	13.54
CQCC-GMM	0.1953	9.87	0.2454	11.04
Bayesian NN	0.0316	1.18	0.0433	1.66
LCNN	0.0516	2.46	0.0600	2.33
Ensemble	**0.0170**	**0.78**	**0.0219**	**0.88**

5. Summary

This work concentrated on achieving good method generalization for replay detection anti-spoofing models. To accomplish this the tested models were pre-evaluated using novel *Attack-Out Cross-Validation* procedure, as a way of estimating their performance against spoofing methods not seen during training. This allowed us to select two potentially robust models - a Bayesian CNN with a small network architecture and an LCNN model trained with a set of regularizing techniques. Results on the cross-validation setup also helped us tune the weights for the combined prediction of the final scores. Such an evaluation setup can be potentially adapted to other settings, where a system's performance on previously unencountered data is crucial. The effectiveness of our cross-validation technique was also confirmed in the ASVspoof 2019 challenge. Finally, the Bayesian network was analyzed further, in order to identify the factors responsible for its good performance. This analysis seems to indicate that a compact model size can be beneficial for generalization in anti-spoofing scenarios. It also raises a question regarding performance of Bayesian neural networks w.r.t. the size of the dataset which can be a potentially interesting direction for future work.

6. References

[1] Z. Wu, T. Kinnunen, N. W. D. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *Interspeech 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, 2015, pp. 2037–2041. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2015/i15_2037.html

[2] T. Kinnunen, M. Sahidullah, H. Delgado, M. Todisco, N. Evans, J. Yamagishi, and K. A. Lee, "The ASVspoof 2017 Challenge: Assessing the Limits of Replay Spoofing Attack Detection," in *INTERSPEECH 2017, Annual Conference of the International Speech Communication Association, August 20-24, 2017, Stockholm, Sweden*, Stockholm, SWEDEN, 08 2017. [Online]. Available: http://www.eurecom.fr/publication/5235

[3] J. Lorenzo-Trueba, J. Yamagishi, T. Toda, D. Saito, F. Villavicencio, T. Kinnunen, and Z. Ling, "The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods," in *Speaker Odyssey 2018 The Speaker and Language Recognition Workshop*, Les Sables d'Olonne, France, 06 2018, pp. 195–202.

[4] T. Kinnunen, K. A. Lee, H. Delgado, N. Evans, M. Todisco, M. Sahidullah, J. Yamagishi, and D. A. Reynolds, "t-DCF: a Detection Cost Function for the Tandem Assessment of Spoofing Countermeasures and Automatic Speaker Verification," in *Speaker Odyssey 2018 The Speaker and Language Recognition Workshop*, Les Sables d'Olonne, France, 06 2018. [Online]. Available: https://hal.inria.fr/hal-01880306

[5] M. Witkowski, S. Kacprzak, P. Zelasko, K. Kowalczyk, and J. Galka, "Audio replay attack detection using high-frequency features," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, 2017, pp. 27–31. [Online]. Available: http://www.isca-speech.org/archive/Interspeech_2017/abstracts/0776.html

[6] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio replay attack detection with deep learning frameworks," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, 2017, pp. 82–86. [Online]. Available: http://www.isca-speech.org/archive/Interspeech_2017/abstracts/0360.html

[7] 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, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0885230816303114

[8] W. Cai, D. Cai, W. Liu, G. Li, and M. Li, "Countermeasures for automatic speaker verification replay spoofing attack : On data augmentation, feature representation, classification and fusion," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, 2017, pp. 17–21. [Online]. Available: http://www.isca-speech.org/archive/Interspeech_2017/abstracts/0906.html

[9] H.-J. Shim, J.-W. Jung, H.-S. Heo, S.-H. Yoon, and H.-J. Yu, "Replay Spoofing Detection System for Automatic Speaker Verification Using Multi-Task Learning of Noise Classes," in *2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. IEEE, 2018, pp. 172–176.

[10] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2348–2356. [Online]. Available: http://papers.nips.cc/paper/4329-practical-variational-inference-for-neural-networks.pdf

[11] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[12] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Networks," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 1613–1622. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045118.3045290

[13] ASVspoof consortium. (2019, 01) ASVspoof 2019: Automatic Speaker Verification Spoofing and Countermeasures Challenge Evaluation Plan. [Online]. Available: http://www.asvspoof.org/asvspoof2019/asvspoof2019_evaluation_plan.pdf

[14] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[15] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, "Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJNpifWAb

[16] X. Wu, R. He, Z. Sun, and T. Tan, "A Light CNN for Deep Face Representation with Noisy Labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018.

[17] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, A. Courville, D. Lopez-Paz, and Y. Bengio, "Manifold Mixup: Better Representations by Interpolating Hidden States," *arXiv e-prints*, p. arXiv:1806.05236, Jun 2018.

[18] T. Inoue, P. Vinayavekhin, S. Wang, D. Wood, N. Greco, and R. Tachibana, "Domestic Activities Classification Based on CNN Using Shuffling and Mixing Data Augmentation," 2018.

[19] H. Delgado, M. Todisco, M. Sahidullah, N. Evans, T. Kinnunen, K. Lee, and J. Yamagishi, "ASVspoof 2017 Version 2.0: Metadata Analysis and Baseline Enhancements," in *Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018.

[20] C.-I. Lai, A. Abad, K. Richmond, J. Yamagishi, N. Dehak, and S. King, "Attentive Filtering Networks for Audio Replay Attack Detection," *arXiv e-prints*, p. arXiv:1810.13048, Oct 2018.

[21] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN Architectures for Large-Scale Audio Classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.09430

[22] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An Ontology and Human-Labeled Dataset for Audio Events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[23] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep Metric Learning with Angular Loss," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 2612–2620. [Online]. Available: https://doi.org/10.1109/ICCV.2017.283

[24] L. van der Maaten, "Accelerating t-SNE using Tree-Based Algorithms," *Journal of Machine Learning Research*, vol. 15, pp. 3221–3245, 2014. [Online]. Available: http://jmlr.org/papers/v15/vandermaaten14a.html

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980