INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Deep-learning-based central African primate species classification with MixUp and SpecAugment

Thomas Pellegrini

IRIT, Université Paul Sabatier, CNRS, Toulouse, France
thomas.pellegrini@irit.fr

Abstract

In this paper, we report experiments in which we aim to automatically classify primate vocalizations according to four primate species of interest, plus a background category with forest sound events. We compare several standard deep neural networks architectures: standard deep convolutional neural networks (CNNs), MobileNets and ResNets. To tackle the small size of the training dataset, less than seven thousand audio files, the data augmentation techniques SpecAugment and MixUp proved to be very useful. Against the very unbalanced classes of the dataset, we used a balanced data sampler that showed to be efficient. An exponential moving average of the model weights allowed to get slight further gains. The best model was a standard 10-layer CNN, comprised of about five million parameters. It achieved a 93.6% Unweighted Average Recall (UAR) on the development set, and generalized well on the test set with a 92.5% UAR, outperforming an official baseline of 86.6%. We quantify the performance gains brought by the augmentations and training tricks, and report fusion and classification experiments based on embeddings that did not bring better results.

Index Terms: automated species classification, primate vocalizations, audio data augmentation

1. Introduction

These last years, the domain of bioacoustics has greatly benefited from the impressive performance breakthroughs of deep learning and deep neural networks (DNNs), alike many other audio-related applications such as speech recognition and audio event detection [1]. Automatic detection of animal vocalizations, besides being a scientific challenge in itself, can be helpful for monitoring biodiversity. Researchers involved in bioacoustics and/or the recent field named ecoacoustics [2] gather ever-growing quantities of *in-situ* recordings that need to be manually analyzed. Automatic tools that label the recordings accurately are very much in demand to ease the time-consuming task of listening to hours of them [3].

A variety of tasks have been addressed with automatic tools: "simply" detecting the absence/presence of animal vocalizations in recordings (bird singing [4], for instance), identifying species of a given animal (the yearly LifeCLEF Bird Identification Task [5], for instance), recognizing individuals in multiple species [6]. All the previous examples dealt with singing birds but many other animals have been considered, such as bats [7], and primates [8, 9, 10].

A number of machine learning techniques have been used in the literature, such as Support Vector Machines either with raw acoustic features or feature representations issued from deep neural networks [9, 11], kernel methods, such as Extreme Learning Machine (ELM) [10], and, of course, a variety of deep neural networks: convolutional and recurrent convolutional neural networks [12], densely connected networks [13], Residual Neural Networks (ResNets) [14], etc.

In this work, we are interested in comparing a number of DNN architectures in the task of distinguishing between four primate species based on their vocalizations: chimpanzees, mandrills, red-capped mangabeys and a mixed group of guenon species. We use the audio dataset provided by the INTERSPEECH 2021 Computational Paralinguistics Challenge [9, 11, 10]. Besides the four primate species of interest, a fifth class is considered in this classification task: natural background noise. In real applications, this "reject" class would be very interesting to have in a classifier if we aim at spotting out primate calls within lengthy audio field recordings.

Besides comparing different models, we also investigate the use of audio data augmentation, as it is key for training networks. We will report significant performance gains using SpecAugment [15] and MixUp [16].

The best system reported by the challenge organizers is based on extracting feature representations from a recurrent autoencoder in an unsupervised fashion and then use these embeddings, named "auDeep", as input to an SVM classifier. We therefore tried to use the auDeep embeddings within one of our best models. Although this improved the auDeep-SVM performance, no gain was obtained with our networks.

2. Models

In this section, the four different model architectures are described. All the models make use of blocks of convolution layers followed by global pooling before a classification head comprised of two fully-connected layers. The difference between the architectures lies in the convolutional blocks' definition.

In the present work, all the models take log-mel spectrograms of 64 coefficients as input. More details on data preprocessing will be given in Section 3.

2.1. Standard CNNs: CNN6 and CNN10

By standard CNNs we mean networks comprised of blocks of a convolution layer, followed by batch-normalization (BN), the rectifier linear unit (ReLU) activation function, and 2×2 average pooling. Dropout of rate 20% is applied after each convolution block. The blocks are followed by, first, an average pooling over the frequency dimension, and second, summation of mean and max-pooling on the time axis. The pooled representations (vectors) are given to a first fully-connected layer (FC) of 512 units, with ReLU, and second to the classification FC layer of dimension the number of classes of interest (five classes).

CNN6 and CNN10 follow this architecture with six and ten layers, respectively. CNN6 consists of four convolution layers with a 5×5 kernel size, and 2×2 padding. CNN10, shown in Table 1, uses kernels of size 3×3 and there are two consecutive convolution layers in each block. CNN6 and CNN10 totalize 4.57 M and 4.95 M parameters, respectively.

Table 1: *CNN10 architecture [17]. The number before @ indicates the number of output channels.*

log-Mel spectrograms 75 frames $\times$ 64 Mel bins
$\begin{pmatrix} 64@3 \times 3 \\ \text{BN, ReLU} \end{pmatrix} \times 2$
Pooling 2×2
$\begin{pmatrix} 128@3 \times 3 \\ \text{BN, ReLU} \end{pmatrix} \times 2$
Pooling 2×2
$\begin{pmatrix} 256@3 \times 3 \\ \text{BN, ReLU} \end{pmatrix} \times 2$
Pooling 2×2
$\begin{pmatrix} 512@3 \times 3 \\ \text{BN, ReLU} \end{pmatrix} \times 2$
Global pooling FC, 512, ReLU FC, 5, Sigmoid or Softmax

2.2. MobileNetV1

MobileNets [18] make use of depthwise separable convolutions to reduce computation costs and compensate with 1×1 kernel sized convolutions, also called point-wise convolutions, in order to exchange information between different channels.

In this work, we used MobileNetV1, comprised of 13 MobileNetV1 convolutional blocks. Each block consists of a depthwise separable convolution, followed by average pooling, BN, ReLU, and then a pointwise convolution with BN and ReLU. The same pooling and FC layers as in the CNNs and ResNet22 are used after the convolutional blocks. In total, this network comprises 4.26 M parameters.

2.3. ResNet22

A ResNet is comprised of convolutional blocks, each of two convolution layers with a 3×3 kernel size, and a shortcut connection between input and output that allows to train efficient deep networks [19]. We used ResNet22, a model comprised of 22 layers: two convolutional layers and a downsampling layer applied on the spectrograms and then eight convolutional blocks with residual connections. Again, the same classification head as the other models is used. The resulting model is much larger than the other architectures, with 62.60 M parameters. Despite this large size and the relatively small number of training samples, the test performance and generalization capability of this model is remarkable, as we shall see hereafter. No training strategy specific to this model was used.

2.4. WideResNet28-2

Wide Residual Networks [20] consist of an initial convolutional layer followed by three groups of residual blocks. After these blocks, follow average pooling and a linear layer that acts as a classifier. The residual blocks, each composed of two "BasicBlocks" with a convolutional layer with 3×3 kernels, Batch Normalization and ReLU, are repeated three times. We used the official implementation available in PyTorch [21] of the Wide ResNet 28-2 architecture, but slightly modified to be more similar to the other models in this study: after the convolution blocks, we use global max- and average-pooling, sum the result and apply two FC layers. It results in a small sized model of about 1.5 million parameters.

3. Data processing and setup

3.1. Feature extraction

The primate vocalization dataset [10] consists of three subsets ("train", "devel" and "test") of comparable size: 6915, 6918 and 6923 audio files of variable duration. We padded all the files to reach three seconds duration, as it corresponds to the largest recording duration of all the files. 64 log-mel spectrograms were extracted with a 1024-sample-long Hanning window and 320 samples hop size. We ran experiments with the original 16 kHz sampled signal and a downsampled 8 kHz version to speed up model comparison. With the 16 kHz version, spectrograms are of size 150×64 in time and frequency.

3.2. Data balancing

The train and development subsets are highly imbalanced regarding the five classes of interest: 50.0%, 32.1%, 2.3%, 12.6%, 3.0% for Background, Chimpanzee, Guenon, Mandrill, and Red-capped mangabeys, respectively.

Training data are input to our models in mini-batches of size 32 samples during training, which is a relatively small size. Given the important imbalanced representation of the classes, it may happen that all the data in a mini-batch may belong to the same sound class Background. This will cause the models to overfit to this class, with more training clips, and underfit to the other sound classes with fewer training clips. To avoid this issue, Kong and colleagues proposed to use a balanced data sampler used during training [17]. With this sampler, audio clips are equally sampled from all the five sound classes when constituting a mini-batch. We will compare the results obtained with this strategy to using a uniform standard sampler.

3.3. Data augmentation

Data augmentation is key to increase the generalization capabilities of DNNs. Nevertheless, finding the right augmentation strategies is a difficult task. In [22], a number of audio-specific augmentations were tested in a systematic way for environmental sound classification: time stretching, pitch shifting, dynamic range compression and background noise addition. The authors show that the impact of using these techniques depends on the sound classes of interest: performance may decrease if the technique is not well suited for a given class.

In this work, several augmentations were tested: time stretching with a rate uniformly sampled in the [0.9, 1.1] interval, pitch shifting by 0 to 3 quarter-tones up or down, MixUp, and SpecAugment. In preliminary experiments, time stretching and pitch shifting did not bring performance improvements. On the contrary, MixUp and SpecAugment revealed very efficient.

MixUp [16] is a successful data augmentation/regularization technique, in which we mix pairs of data samples (images, audio clips, etc.). If x_1 and x_2 are two different input samples (spectrograms in our case), and y_1, y_2 their respective one-hot encoded labels, then the mixed sample and target are obtained by a simple convex combination:

$$x^{\text{mix}} = \lambda x_1 + (1-\lambda)x_2$$
$$y^{\text{mix}} = \lambda y_1 + (1-\lambda)y_2$$

where λ is a scalar sampled from a symmetric Beta distribution at each mini-batch generation:

$$\lambda \sim \text{Beta}(\alpha, \alpha)$$

where α is a real-valued hyper-parameter to tune (always smaller than 1.0 in our case).

SpecAugment [15] is an occlusion augmentation technique, applied onto the log-mel spectrograms. In our case, we randomly choose to mask zero, one or two stripes both in the time and frequency axes. The width of the strides is also chosen randomly with a maximum of 16 and 8 bins in time and frequency, respectively. SpecAugment is applied at mini-batch level, meaning that the same random strides are masked in all the samples of a given mini-batch. SpecAugment has been originally proposed in automatic speech recognition but it has been rapidly used with success also for other audio-related tasks, such as audio tagging.

3.4. Experimental setup

The models were trained to optimize binary cross-entropy, when MixUp was used, and categorical cross-entropy otherwise. In all our experiments, we used the Adam [23] optimizer to train the models with a learning rate of 1e-3. The size of the minibatches was set to 32. We trained our model for 60k iterations. Except WideResNet, all the model architectures are the ones proposed in [17] for multilabel audio event detection and we largely used their open-sourced implementation[1]. Our PyTorch [21] code is available[2].

We used Exponential Moving Average (EMA) to obtain our final models: during the training of any model, we create a separate model which is the EMA variant of the model being trained. We used a 0.995 decay and EMA updates were initiated after 15k training iterations.

4. Results

In this section, we report results in terms of mean average precision (mAP) and unweighted average recall (UAR). UAR is the metric used in the Primate sub-challenge, occurring in the framework of the yearly Paralinguistic Compare challenges.

4.1. Comparison between models

Table 2 shows the results obtained with the different models on the development subset. As a global comment, all the DNNs outperformed the 84.6% UAR reported in the official challenge paper [11]. The best model is CNN10-16k, the 10-layer standard CNN, that achieved a 93.6% UAR on the devel subset and 92.5% on the official test set. CNN10-16k is closely followed by ResNet22-16k. CNN10-16k's size is an order of magnitude smaller than ResNet22, thus, besides being slightly more accurate, it is also much more preferable in terms of computation. CNN6, the 6-layer standard CNN is ranked third closely followed by MobilenetV1. The least performing model was WideResNet28-2. This ranking between architectures should be considered with caution because we used the same learning hyper-parameters for all models. It may happen that the 1e-3 learning rate is not the best one for all the models, for instance. Nevertheless, the fact that a standard CNN is the best model in these experiments, is in line with the results obtained by Kong and colleagues on AudioSet [17]. We tested CNNs with more layers (CNN14) on our task, but CNN10 proved to be better, probably because CNN14 is too deep to be correctly trained with our small dataset.

[1]https://github.com/qiuqiangkong/audioset_tagging_cnn

[2]https://github.com/topel/ComParE2021_PRS

Table 2: *Comparison between models. mAP: average precision, UAR: unweigthed average recall. 8k, 16k: audio sampling rate.*

	devel		test
setting	mAP (%)	UAR (%)	UAR (%)
ELM [10]	60.3	61.0	70.7
auDeep-SVM [11]	N/A	84.6	86.6
CNN6-8k	92.9	88.2	-
CNN10-8k	95.4	91.9	90.6
MobileNetV1-8k	91.4	88.0	-
ResNet22-8k	93.8	91.2	89.1
WideResNet28-2-8k	92.8	87.7	-
ResNet22-16k	95.7	92.6	-
CNN10-16k	**97.0**	**93.6**	**92.5**

In Table 2, we also report three UAR values obtained on the official challenge test set. CNN10-16k reached a 92.5% UAR on the test set, showing that it generalized well on these new data. It outperformed the ELM model [10] at 70.7%, and the 86.6% auDeep-SVM baseline provided by the challenge organizers [11].

Figure 1 shows the confusion matrix on the devel subset, obtained with our best model. The best recognized classes are guenon (G) and mandrills (M) with about 97% accuracy. Background (B) has the lowest accuracy, about 89%, and most confusions are made with samples of the chimpanzee class (C), and to a less extent with the other primate species. Finally, Red-capped mangabeys (R) are confused with Mandrills in 3.3% of the cases.

4.2. Ablation study

Table 3 shows the impact of several components: balanced sampler, data augmentation and EMA. This ablation study was performed with ResNet22-8k and CNN10-16k.

The first two lines compare the use of a random sampler (no-bal) and a balanced sampler (bal). Using a random sampler is worse than a balanced one in UAR, as expected. In the case of ResNet22, there is a two points difference in mAP and 4 points in UAR. This is because the background class is better recognized with no-bal, and it is by far the most represented class in the dataset (50% of the samples).

The next three lines show the impact of data augmentation: specAugment alone (specA), MixUp alone and specAugment and MixUp together. We can see that both techniques boost mAP and UAR significantly: about 3 points absolute in mAP and 5.5 points in UAR for ResNet22, or instance. Finally using both methods together gave the best results for both models.

The last line in the table shows the results of the counterpart of the EMA model of the preceding line. EMA shows slightly better results than their non-EMA counterparts. This tendency was observed with all the architectures.

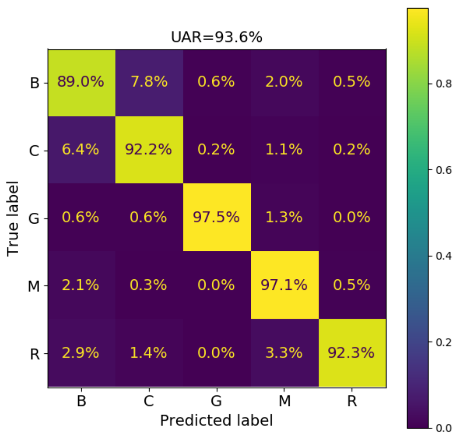

Figure 1: *Confusion matrix on the devel subset, obtained with our best model CNN10-16k. B: Background, C: Chimpanzee, G: Guenon, M: Mandrille, R: Red-capped Mangabeys.*

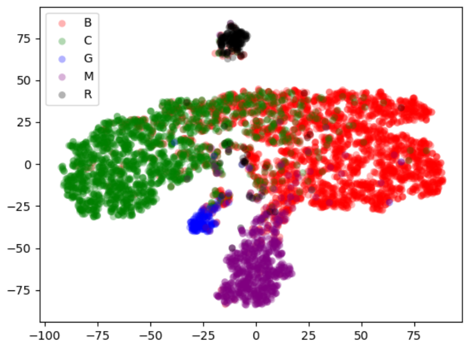

Figure 2: *2-d representations obtained with t-SNE on the embeddings obtained with CNN10-16k on the devel samples.*

5. Complementary experiments

In this section, we briefly describe complementary experiments that did not bring improvements, at least in UAR.

CNN10 embeddings and SVM. The challenge organizers reported their best results using Support Vector Machines with a linear kernel applied onto embeddings obtained with a deep recurrent neural network named auDeep: 84.6% and 86.6% of the development and test subsets respectively. In a similar fashion, we extracted CNN10 embeddings outputted from the penultimate fully-connected layer in CNN10. Figure 2 shows a two-dimensional representation of the embeddings obtained with t-SNE [24] on the development subset. We can observe that the five classes are mostly well separated in the plane, Background (in red) and Chimpanzee (in green) to a less extent though. Linear SVMs reached a 88.8% UAR on the development set.

Fusion between CNN10 and ResNet22. Since CNN10 and ResNet22 achieved very similar mAP and UAR values, we experimented an intermediate fusion: we defined a “ResNet22-CNN10” model that combines CNN10-8k and ResNet22-8k as two parallel branches to extract embeddings from their penultimate fully-connected layer. Their embeddings, of size 512 (CNN10) and 2048 (ResNet22) are then simply concatenated and given as input an FC classification layer. Both CNN10 and ResNet22 branches were initialized with the weights of the best models reported above. We tried to freeze or not these branches, and we also tried with and without augmentation (specAugment and/or MixUp). Freezing the branches and using both augmentations led to the best results: 91.6% UAR and 96.2% mAP. This UAR value is below that of the CNN10-8k alone. Nevertheless, the 96.2% mAP is slightly larger than the CNN10-8k 95.4% value.

Table 3: *Ablation study mAP(%)/UAR(%) results using CNN10-16k on the development subset. bal: balanced data sampler, no-aug: no augmentation at all, specA: SpecAugment.*

setting	ResNet22-8k	CNN10-16k
no-bal, no-aug	87.9/78.7	94.7/90.9
bal, no-aug	89.8/82.6	94.7/92.1
bal, specA	92.6/88.1	94.1/92.1
bal, MixUp	92.4/88.8	96.0/92.5
bal, specA+MixUp	**93.8/91.2**	**97.0/93.6**
_ w/o EMA	93.1/90.0	96.7/93.0

6. Conclusions

We reported experiments aiming to classify primate vocalizations into four primate species of interest, plus a background category with forest sound events. We compared standard DNN architectures: standard deep convolutional neural networks (CNNs), MobileNets, ResNets and WideResNets.

To tackle the relative small size of the training dataset, less than seven thousand audio files, the data augmentation techniques SpecAugment and MixUp proved to be very useful. Against the very unbalanced distribution of the classes, we used a balanced data sampler, which was efficient with a ResNet but not with our standard CNN. An exponential moving average of the model weights allowed to get slight further gains. The best model was a standard 10-layer CNN, comprised of about five million parameters, that achieved a 92.5% UAR on the test subset. We quantified the performance gains brought by the augmentations and training tricks, and report fusion and classification experiments based on embeddings that did not bring better results in UAR at least.

Most confusions occur between Background and all the primate species, thus, a hierarchical system would be worth trying. One can also imagine a single model but trained on both objectives.

7. Acknowledgements

This work was partially supported by the French ANR agency within the LUDAU project (ANR-18-CE23-0005-01) and the French ”Investing for the Future — PIA3” AI Interdisciplinary Institute ANITI (Grant agreement ANR-19-PI3A-0004). We used HPC resources from CALMIP (Grant 2020-p20022).

8. References

[1] D. Stowell, “State of the art in computational bioacoustics and machine learning: How far have we come?” *Biodiversity Information Science and Standards*, vol. 3:e37227, 2019.

[2] J. Sueur and A. Farina, "Ecoacoustics: the ecological investigation and interpretation of environmental sound," *Biosemiotics*, pp. 1–10, 2015. [Online]. Available: http://dx.doi.org/10.1007/s12304-015-9248-x

[3] P. Guyot, A. Eldridge, Y. C. Eyre-Walker, A. Johnston, T. Pellegrini, and M. Peck, "Sinusoidal modelling for ecoacoustics," in *Proc. INTERSPEECH 2016*, San Francisco, 2016, pp. 2602–2606. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2016-361

[4] D. Stowell, M. D. Wood, H. Pamuła, Y. Stylianou, and H. Glotin, "Automatic acoustic detection of birds through deep learning: the first bird audio detection challenge," *Methods in Ecology and Evolution*, vol. 10, no. 3, pp. 368–380, 2019.

[5] H. Goëau, H. Glotin, W.-P. Vellinga, R. Planqué, and A. Joly, "LifeCLEF Bird Identification Task 2016: The arrival of Deep learning," in *CLEF: Conference and Labs of the Evaluation Forum*, vol. CEUR Workshop Proceedings, no. 1609, Évora, Portugal, Sep. 2016, pp. 440–449. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01373779

[6] D. Stowell, T. Petrusková, M. Šálek, and P. Linhart, "Automatic acoustic identification of individuals in multiple species: improving identification across recording conditions," *Journal of the Royal Society Interface*, vol. 16, no. 153, 2019.

[7] Y. Prat, M. Taub, and Y. Yovel, "Everyday bat vocalizations contain information about emitter, addressee, context, and behavior," *Scientific Reports*, vol. 6, no. 1, pp. 1–10, 2016.

[8] S. Heinicke, A. K. Kalan, O. J. Wagner, R. Mundry, H. Lukashevich, and H. S. Kühl, "Assessing the performance of a semi-automated acoustic monitoring system for primates," *Methods in Ecology and Evolution*, vol. 6, no. 7, pp. 753–763, 2015.

[9] M. Versteegh, J. Kuhn, G. Synnaeve, L. Ravaux, E. Chemla, C. Cäsar, J. Fuller, D. Murphy, A. Schel, and E. Dunbar, "Classification and automatic transcription of primate calls," *The Journal of the Acoustical Society of America*, vol. 140, no. 1, pp. EL26–EL30, 2016. [Online]. Available: https://doi.org/10.1121/1.4954887

[10] J. A. Zwerts, J. Treep, C. S. Kaandorp, F. Meewis, A. C. Koot, and H. Kaya, "Introducing a Central African Primate Vocalisation Dataset for Automated Species Classification," *arXiv preprint arXiv:2101.10390*, 2021.

[11] B. W. Schuller, A. Batliner, C. Bergler, C. Mascolo, J. Han, I. Lefter, H. Kaya, S. Amiriparian, A. Baird, L. Stappen, S. Ottl, M. Gerczuk, P. Tzirakis, C. Brown, J. Chauhan, A. Grammenos, A. Hasthanasombat, D. Spathis, T. Xia, P. Cicuta, M. R. Leon J. J. Zwerts, J. Treep, and C. Kaandorp, "The INTERSPEECH 2021 Computational Paralinguistics Challenge: COVID-19 Cough, COVID-19 Speech, Escalation & Primates," in *Proc. INTERSPEECH*. Brno, Czechia: ISCA, 2021, to appear.

[12] T. Grill and J. Schlüter, "Two convolutional neural networks for bird detection in audio signals," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2017, pp. 1764–1768.

[13] T. Pellegrini, "Densely connected cnns for bird audio detection," in *Proc. EUSIPCO*, Kos, 2017, pp. 1734–1738.

[14] S. Kahl, C. M. Wood, M. Eibl, and H. Klinck, "Birdnet: A deep learning solution for avian diversity monitoring," *Ecological Informatics*, vol. 61, p. 101236, 2021.

[15] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple augmentation method for automatic speech recognition," in *Proc. INTERSPEECH*, Graz, 2019, pp. 2613–2617.

[16] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," in *Proc. ICLR*, Vancouver, 2018.

[17] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[18] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2017.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *proc. NeurIPS*, 2019, pp. 8026–8037. [Online]. Available: http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[22] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2017.

[24] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.