Music Genre Classification with Convolutional Neural Networks Using Data Augmentation Techniques

Emerson Kiefer
University of Massachusetts Amherst
egkiefer@umass.edu

Raymond Li
University of Massachusetts Amherst
rcli@umass.edu

Eugene Mak
University of Massachusetts Amherst
emak@umass.edu

Abstract

Music Genre Classification is an information retrieval task used in popular applications such as music recommendation and playlist generation. We can employ deep learning to do genre classification by converting audio data to images, specifically waveform spectrograms. By working with spectrogram data, we can use Convolutional Neural Networks to classify data to a particular genre. In this paper, we investigate expanding on this concept by experimenting with data augmentations. We will employ data augmentations on either the raw audio data or the spectrograms. We experiment with Add Noise, Pitch Shift, and Mixup augmentations, which has been shown from previous literature to perform well for deep learning with audio classification. We use the GTZAN dataset, a widely used dataset for genre classification. We propose a classification method that could exceed the performance of the best deep learning model for music genre classification by running the model on data with various high performance improvement augmentations.

1. Introduction

Music genre classification is an information retrieval task where the objective is to classify a song based on its data contents. It is an important task, used for popular applications such as music recommendation or playlist generation. However, music genre classification can be challenging, in large part because classifying music is not as easily well-defined and immediately recognizable as other classification tasks such as objects in images. To address this challenge, we aim to train a system to classify music by genre computationally.

One straightforward method for this classification would be by training a convolutional neural network on spectrogram images derived from audio data of songs. This method can be effective, but it still has much room for improvement. We aim to improve performance by both tweaking the architecture and hyperparameters, as well as investigating ways to augment the data itself. Specifically, we will focus on moving beyond just using the spectrogram provided and instead augment the audio data through various transformations, such as time shifting, changing speed/pitch, or noise injection. These augmentations will hopefully show an improvement in performance beyond just using the input data spectrograms by training a neural network on these augmented inputs instead.

By reviewing previous experiments through prior literature, we hope to discover augmentations that will result in an increase in performance compared to a naive baseline standard classification strategy. By combining this approach with the highest-performing neural network architecture seen in previous literature, we hope to achieve a significant relative improvement after applying augmentations compared to without.

2. Problem Statement

For this project, we use the GTZAN dataset, which has been compared to MNIST as the analogous standard dataset for audio music classification. The current best-performing CNN models predict GTZAN music genres with an accuracy of around 83%[1]. We first hope to replicate the best-performing models, with the goal of achieving even higher accuracy using augmentations. Acknowledging the limitations in our compute resources, storage, and project scope, we will implement the best-performing existing CNN that we can feasibly implement before any augmentations as a control measure of performance.

We will also compare our best-performing model's accu-

racy to other modern, high-performance CNN models such as ResNet18 to demonstrate whether or not our specific approach to the genre classification problem has outperformed a well-known, high-performing CNN.

This best-performing model, as well as any other models we test, will be trained and validated using a subset of the GTZAN dataset. Testing instances will be excluded from the training process. To achieve higher performance, we modify the training dataset by performing a variety of data augmentations. We will compare the model's performance without augmentations with the model's performance with one or many of the augmentations to see if any of the augmentations noticeably improve performance.

According to Wei et al. [3], certain data augmentations could improve the generalization of a deep learning network. As such, we expect augmentations to show some sort of appreciable improvement compared to the base model. However, we also understand that the augmentations could be less effective for this particular classification task, and as such we could also foresee the performance not increasing by any significant amount, or even lowering performance if, for example, the augmentations are not suitable for this dataset, or possibly if the augmentations are not compatible with the models we use.

We will evaluate all of the models by comparing their accuracies on the testing set of spectrograms from the GTZAN dataset. We will also display a confusion matrix for each model to easily visualize how our model performs, and compare its exact behavior with the other models.

3. Technical Approach

We implement a convolutional neural network with an identical architecture to the CNN defined in the paper *Convolutional Neural Networks Approach for Music Genre Classification*[1], which boasts an 83.3% accuracy based on majority voting. This model is the highest performing model we could find at this current time on GTZAN.

To achieve their performance, they ran the experiment for 13 hours with 100 epochs. Due to both time and compute limitations, we drastically reduce the number of epochs to 10 in order to limit the time and compute bandwidth needed to run each model. In exchange for lower raw performance, doing this enables us to test a higher variety of models, allowing us to easily focus on comparing relative performance. We keep the number of epochs consistent for each model to ensure fair comparison.

The model consists of 5 convolutional layers, each of which uses ReLU, Maxpooling, and Dropout. The final layer passes its output to a fully connected layer, and the predicted class is determined by majority vote. For the convolutional layers, this model uses a kernel of shape (3, 3), a stride of (1, 1), a max pooling layer of (2, 2), and a dropout of 0.5 as its hyperparameters. The authors used 30% of the GTZAN dataset for testing and the remaining 70% for training. This model will serve as the reference model and will henceforth be referred as the baseline model. Our goal is to use data augmentation and make modifications to the hyperparameters and model architecture to surpass the baseline model's performance.

In addition to this model, we find that certain augmentations have the potential to improve the overall performance of the model. Based on *A Comparison on Data Augmentation Methods Based on Deep Learning for Audio Classification*[3], audio data augmentations have the potential to reduce overfitting and improve generalized performance of deep learning networks. These augmentations can have different properties; for instance, augmentations can modify either the raw audio or the spectrogram, preserve the label or modify it, linearly or nonlinearly interpolate samples, etc.

In their comparison of data augmentations, Mixup demonstrated the highest performance percentage improvement. Mixup linearly interpolates samples to generate new samples, constructing new soft labels for the new samples. Labels overall are converted to one-hot encoding, and the increase in training data improves generalization of the model. Thus, due to its high performance, we test Mixup applied on the GTZAN dataset along with the model from Cheng et al [1].

We also test pitch shift and add noise. Pitch shift changes the waveform by a certain user-defined level of semitones, and add noise adds Gaussian noise to smooth the input space and make training easier. From Wei et al, these augmentations demonstrate appreciable performance improvement while also being relatively simple augmentations to implement. We test all three augmentations independently to observe which of the augmentations show the greatest improvement.

Along with these augmentations, we are interested in testing the effectiveness of positional encoding as an architectural modification. One key difference between images and spectrogram images is the importance of position in the spectrogram. Convolutional neural networks strive to be translation-invariant, meaning that translating an object in the image should not have an impact on performance. Intuitively this makes sense for image classification and detection problems because for each of these problems the location of the object generally shouldn't impact the confidence of its classification. While this works for images, the positional information in spectrograms carries far more weight. The value of a point (time, frequency) is determined by the amplitude of that frequency at that time. The translation-invariance of a traditional CNN means that a CNN trained on spectrograms would generally ignore time and frequency. Our intuition is that this may harm performance because frequency is critical to the way humans process, understand, and classify music. We experiment with

positional encoding as an additional modification that we believe may improve our model.

4. Results

4.1. Arbitrary Model

Based on an example tutorial model provided by Won et al. [4], we first constructed and established a simple arbitrary CNN model to ensure we have a working framework for testing different CNNs on GTZAN. As a point of comparison, this model also allows us to ensure that the baseline model is indeed high performing and working as intended. We tested a simple 5-layer CNN, with each layer including batch normalization, max-pooling, and dropout. The convolutional depth gradually doubles in size for every two layers, initially starting with 16 channels. The model finishes off with a series dense layers: one linear connected layer that returns the same output size, a batch normalization layer, another linear layer that downsamples to the number of classes, dropout with $p = 0.5$, and ReLU. The loss function used is cross-entropy loss, so the output scores are processed with softmax.

The exact details of the convolution layers are the following:

- Layer 1 with 16 output channels and max pooling of (2,3).
- Layer 2 with 16 input and output channels and max pooling of (3,4).
- Layer 3 with 16 input and 32 output channels, and max pooling of (2,5).
- Layer 4 with 32 input and output channels, and max pooling of (3,3).
- Layer 5 with 32 input and 64 output channels, with max pooling of (3,4).

The example model also preprocesses the data by converting the inputs to mel spectrograms, which are fast to compute and have shown fast performance. The model also uses the Adam optimizer with a learning rate of 0.001. For the other models, we will use optimizers described in previous literature to determine the best performing optimizer and learning rate. Training with just 10 epochs, we have achieved an accuracy of approximately 42.7%. Thus, this accuracy becomes our reference for an arbitrary model that should theoretically perform worse than the baseline model, and possibly worse than the other models with augmentations. This model was also tested on data without any kind of data augmentation, allowing us to observe if the baseline model with augmentations possibly decreases performance to the point of even worse performance than an arbitrary model. We expect to use this performance data point as a reference to see how effective certain augmentations can be, and combine the most effective augmentations with the highest performing CNN architecture to test on GTZAN.

From Figure 1, we can see that our model is inconsistent with how well it performs certain genres. To a certain degree, we may have to expect this outcome, due to the fact that certain genres of music are harder to distinguish than others. From the graph, notably we can see that our model classifies classical and jazz the best, though it misclassifies many of blues music as jazz. It also notably incorrectly classifies many songs as disco. Observing this confusion matrix can give us some insight into the most challenging genres to classify, and give us clues into specific ways we should implement improvements to our models that address the flaws we observe here.

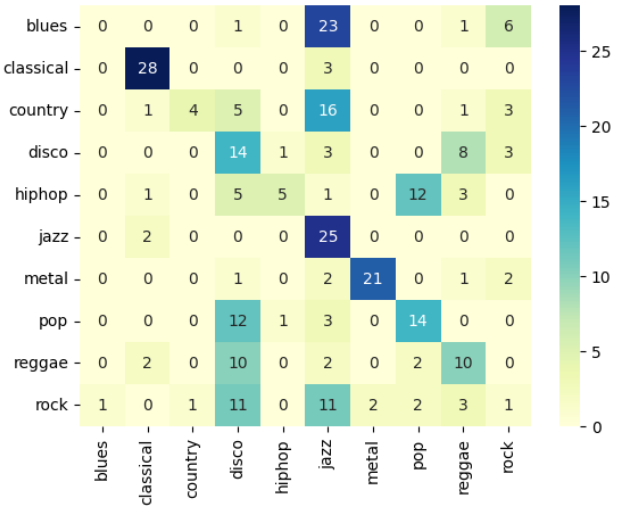

Figure 1. Confusion matrix of the arbitrary model.

4.2. Baseline Model

This baseline model is the model shown in *Convolutional Neural Networks Approach for Music Genre Classification*[1]. Compared to the arbitrary model, this model does not use batch normalization for the layers, and only has one dense fully connected layer after the convolution layers which outputs to the number of classes. However, each layer also has a dropout layer with $p = 0.5$ and uses the ReLU activation function. This model also uses the Adam optimizer, which we set the learning rate to 0.001.

The exact details of the convolution layers are the following:

- Layer 1 with 128 output channels and max pooling of (2,2).
- Layer 2 with 128 input channels, 64 output channels and max pooling of (2,2).
- Layer 3 with 64 input and 32 output channels, and max pooling of (2,2).
- Layer 4 with 32 input and 16 output channels, and max pooling of (2,2).
- Layer 5 with 16 input and 8 output channels, with max

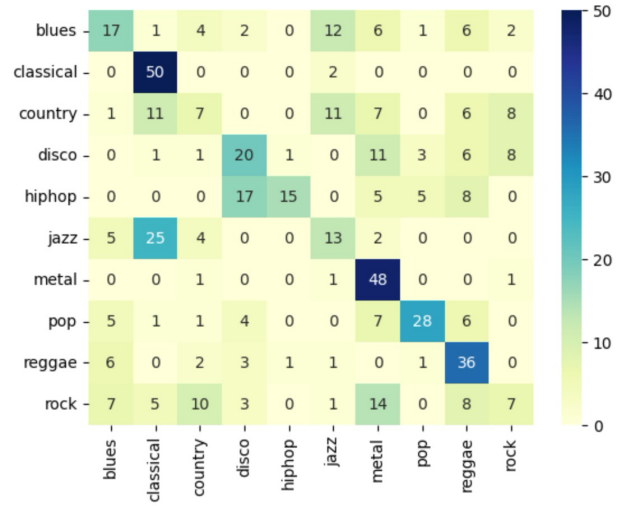

Figure 2. Confusion matrix of the baseline model.

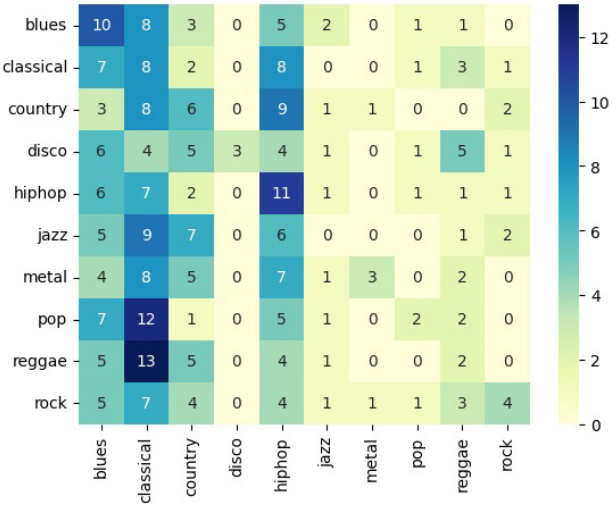

Figure 3. Confusion matrix of ResNet18.

pooling of (2,2).

In contrast to the arbitrary model, this model and all future models uses a preprocessed melspectrogram version of GTZAN rather than converting during the forward pass. This allows us to minimize computation during training and testing, while also allowing us to easily modify this dataset with augmentations, particularly the augmentations that modify the spectrogram data.

With 10 epochs, the baseline model features 47.07% accuracy, an improvement compared to the arbitrary model. Though this accuracy is lower than the accuracy in the paper, this is to be expected due to the lower number of epochs. Instead, we will use this metric to compare relative performance improvements so that we can observe how well augmentations perform with the model, and infer its performance when run on all 100 epochs.

From Figure 2, we can see that the baseline model is particularly good at classifying certain genres. Specifically, the baseline model correctly classifies classical and metal the most out of all the genres. However, we can also see that it incorrectly classifies many jazz samples as classical. This confusion could possibly be explained by the fact that both are instrumental genres. Another example is with metal and rock, as it considers many rock songs as metal; both genres heavily involve electric guitars, and certain metal songs could be considered rock by even an average person. In general, the genres that are confused appear to be relatively grounded in reality.

4.3. ResNet18

ResNet18 is a well known deep learning model from *Deep Residual Learning for Image Recognition*[2]. It is an 18-layer Convolutional Neural Network trained on ImageNet. ResNet18 is known to be a well performing model, and is implemented in the Pytorch library. Due to its performance on ImageNet, we expected ResNet18 to perform somewhere between the arbitrary model and the baseline model.

However, with 10 epochs, ResNet18 on GTZAN had an average accuracy of 16.3%, far below any other model we've tested. The performance of this model seems to indiciate that it is not able to generalize sufficiently at all for this dataset. We hypothesize that the model's architecture and beahvior is better suited to images of visual objects, rather than the more particular spectrogram images, so it cannot properly distinguish spectrograms. Due to the extremely poor performance of ResNet18, we focus on augmenting with only the baseline model.

From Figure 3, we can see that the model does not seem to be able to generalize on the dataset. In other words, it is unable to consistently classify any sort of data, as we can see that it incorrectly classifies nearly every genre as either blues classical, country, or hiphop. Thus, even with a variety of different category of spectrograms, it considers many of them to be part of the same genre, implying that the model is unable to properly distinguish spectrograms.

4.4. Baseline Model With Gaussian Noise

The Add Noise augmentation adds Gaussian Noise to all audio samples, applied to the raw audio before spectrogram conversion. Gaussian Noise smooths the input and theoretically improves learning for the model. Noise amplitude σ is a hyperparameter, with small values increases difficulty in disturbing the inputs and large values increasing difficulty of learning. From Wei et al.[3], an acceptable range of σ is $[0.001, 0.015]$. We use a σ value of 0.001.

New samples after the augmentation is generated with

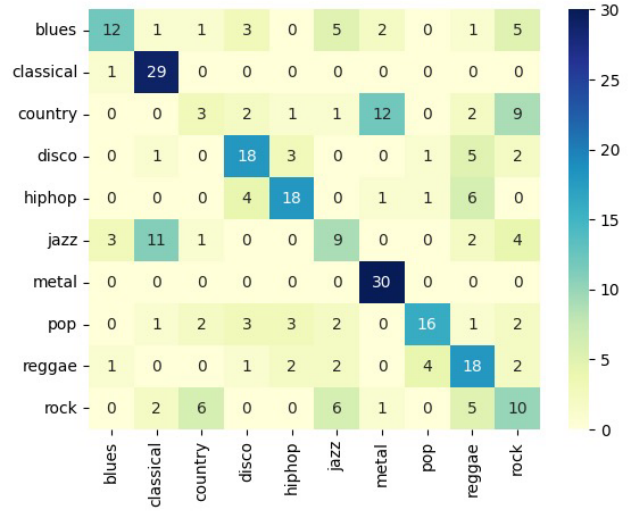

Figure 4. Confusion matrix of the baseline model with Gaussian Noise.

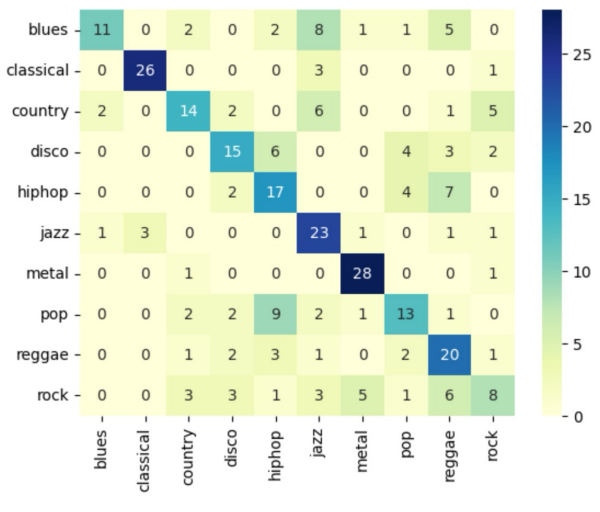

Figure 5. Confusion matrix of the baseline model with Pitch Shift.

the following expression:

$$x(t) = x(t) + \sigma n(t) \tag{1}$$

The accuracy of the model with noise is 54.33%. This is a significant improvement from baseline. We suspect that this is in part due to the generation of new samples. With more data to train on, the model seems to be able to generalize better than without the new samples.

From Figure 4, we can see that it is able to classify classical and metal better than any other genre, much like the baseline. Also similar to baseline, it misclassifies many jazz examples as classical. However, it appears to correctly classify more samples correctly, as indicated by the frequency heatmap indicator on the diagonal. These results shows that the added samples have a positive impact, and that adding noise can indeed improve music genre classification.

4.5. Baseline Model With Pitch Shift

The Pitch Shift augmentation simply pitches the waveform by the user-controlled hyperparameter n_steps of semitones without changing tempo. This augmentation modifies the raw audio. Wei et al. [3] defines the acceptable range of n_steps to be from $[-4, 4]$.

The accuracy of this model is 58.33%. This augmentation demonstrates even higher performance than compared to Gaussian noise, although the performance of the two augmentations are comparable. Similar to the earlier augmentation, the addition of new samples, and thus a greater training size, likely contributes to the increase in performance. However, the performance of Pitch Shift specifically is still appreciably higher than Gaussian Noise. We hypothesize that by training on two different spectrograms that belong to the same genre but different amplitudes, it trains the model to determine the genre based on the shape of the waveform itself, possibly leading to greater accuracy.

From Figure 5, we can see that it also classifies similar to the baseline model. However, the most important difference is that pitch shift has significantly reduced the amount of jazz misclassifications. We hypothesize that pitch shift may be related to the nature of the jazz genre, or that in other words, jazz itself may be particularly sensitive to pitch. Alternatively, the pitch of classical and jazz may have been too similar to each other, causing the confusion with the other models. Regardless, by shifting the pitch, we positively impacting samples in the jazz category to properly distinguish it from other genres.

4.6. Baseline Model With Mixup

The Mixup augmentation linearly interpolates two random samples to return a new sample, with λ being a hyperparamter to control the mixed factor. Mixup does not preserve labels, unlike Add Noise and Pitch Shift. Instead, new soft labels are constructed by mixing the two labels, and labels are converted into a one-hot encoding vector. The newly generated soft labels are then considered to be part of certain categories of the raw input data to return the particular class/genre. Becuase it increases the training data, it can control the models complexity. Thanks to the increase of training data, Mixup can improve generalization of the model[3].

The new samples and labels are generated with the following expressions:

$$\widetilde{x} = \lambda x_i + (1 - \lambda) x_j \tag{2}$$

$$\widetilde{y} = \lambda y_i + (1 - \lambda) y_j \tag{3}$$

Model	Augmentation	Accuracy	Performance Improvement from Baseline (%)
Arbitrary	–	42.7	0.91
Baseline	–	47.0	1.00
ResNet18	–	16.3	0.35
Baseline	Noise	54.3	1.21
Baseline	Pitch Shift	58.3	1.24
Baseline	Mixup	47.7	1.01

Table 1. Summary of performance for all models run.

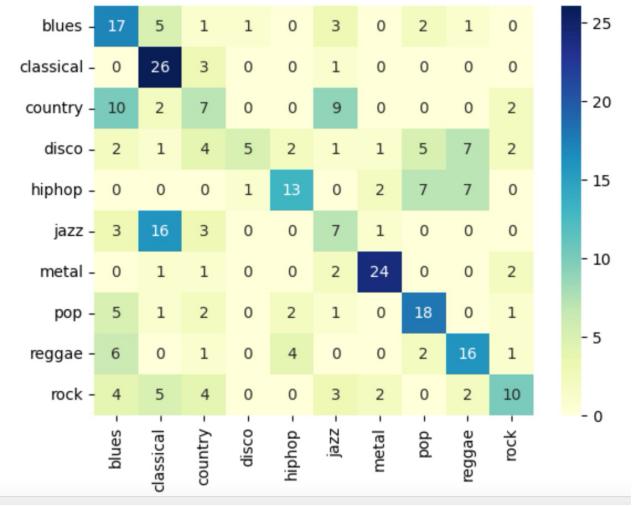

Figure 6. Confusion matrix of the baseline model with Mixup.

The accuracy of this model with Mixup is 47.66%.

Since Mixup was the highest performing augmentation, we hypothesize that the performance improvement is not as high as we anticipated due to compute limitations; as a concession, we had to tune the hyperparameter to a lower value in order to satisfy our compute constraints. Doing this may have hindered the effectiveness of the augmentation. We suspect that if we were able to run the model longer for more epochs at a higher hyperparameter value that generates more samples, we could possibly reach a higher potential for the augmentation and achieve much greater performance.

From Figure 6, we can see that the performance is comparable to the baseline model, with the high performance of classical and metal while also showing the confusion of jazz samples as classical.

4.7. Summary of Results

From Table 1, we observe that all three augmentations consistently show a noticeable improvement from both the arbitrary model and the baseline model without augmentations.

5. Conclusion

In our work, we have successfully constructed and tested several models for music genre classification using GTZAN. These models varied in effectiveness. We showed that ResNet18 does not appear to perform well at all with this dataset, featuring the lowest accuracy of all the models. Though it is possible that ResNet could possibly be tuned to perform well at this classification task, we conclude that ResNet is not suitable for this task as well as other deep learning models. We showed that our baseline model on GTZAN does perform better than other arbitrary deep learning models, even if its performance is reduced compared to the performance seen in the paper. Most significantly, we have shown that augmentations have the potential to give an appreciable performance improvement for this task. Specifically, we demonstrated that the baseline model combined with augmentations results in a appreciable performance improvement, requiring no change to model architecture or training time. Due to the ease of implementation and relatively low penalty in increased performance time, we believe augmentations can a simple yet effective way to increase performance for this classification task.

For future work, we hope to evaluate these models for a greater number of epochs to align with the results seen in previous literature. We also believe that we could do further experiments with each augmentations' hyperparameters. Lastly, we would be interested in stacking the augmentations to see if using multiple augmentations at once results in even greater accuracy. We would also be interested in researching why these augmentations are effective at all. According to Wei et al. [3], they too hope to further explore reasons why augmentations are effective. In doing so, we could intelligently choose or even design augmentations that would be most effective for this particular classification task. With more time and compute resources, we believe this work could be much further expanded upon in the future and show greater results than previously seen.

Go to this link to access our video: https://drive.google.com/file/d/1xmnrMt81FM-NzVKNQW5kxYorNKShIW0r/view?usp=sharing

References

[1] Yu-Huei Cheng, Pang-Ching Chang, and Che-Nan Kuo. Convolutional neural networks approach for music genre classification. In *2020 International Symposium on Computer, Consumer and Control (IS3C)*, pages 399–403, 2020. 1, 2, 3

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 4

[3] Shengyun Wei, Shun Zou, Feifan Liao, and Weimin Lang. A comparison on data augmentation methods based on deep learning for audio classification. *Journal of Physics: Conference Series*, 1453:012085, 2020. 2, 4, 5, 6

[4] Minz Won, Janne Spijkervet, and Keunwoo Choi. *Music Classification: Beyond Supervised Learning, Towards Real-world Applications*. https://music-classification.github.io/tutorial, 2021. 3