12/07/2024

Liam Laidlaw

Brandon Rose

Mario Slaybe

Dr. Huelsman

**A Comparison of CNN, RNN and FCNN for Music Genre Classification**

**Abstract**

This project discusses different strategies for classification of music using deep neural networks. Discussed are various network infrastructures including traditional feed forward networks, convolutional networks, recurrent networks, and convolutional recurrent neural networks, as well as advantages and disadvantages for each in this application. Using the PyTorch framework, each of these implementations were tested in order to evaluate their efficacy as music genre classifiers.

1. **Introduction**

There are a variety of neural networks each good at performing certain tasks; fully connected networks (FCNN) at learning complex relationships between inputs, convolutional neural networks (CNN) are good at image classification, and recurrent neural networks (RNN) at handling sequential data. Depending on the use case these differences are important for determining which to use in the situation. While there are a large variety of other networks, some even a combination of the three mentioned above, our goal is to test the performance and capabilities of each three when it comes to classification of music genres. For our training we used the GTZAN dataset which features three-second and thirty-second clips of music which comes in the form of csv with fifty-eight attributes and ten different genres. In our approach we created a FCNN, CNN, and RNN to test their effectiveness and compare them against each other. The FCNN was used as the baseline for testing against. While there has been a variety of work done on music genre classification, especially with the GTZAN dataset, we wanted to explore the differences in models ourselves and see how our models compare to others. We expected the CNN to perform the best and the RNN to perform the worst when working on this dataset.

Before implementing our own, we first looked through different research papers to see what had already been determined. Doing this helped point our efforts in the right direction. From the research of others, we found that convolution networks seem to be the most effective in classifying music genres.

Through analysis of this research, it was determined that a combination of different neural network types increases accuracy. In “Music Genre Classification using Convolutional Recurrent Neural Network,” A combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) were used. The research in this paper found this combined model that could look at both spatial and temporal was superior to more traditional approaches. The results of this paper inspired us to attempt different approaches and see if they can produce more accurate results.

**2. Literature Review**

The GTZAN dataset was first released in 2002 by George Tzanetakis and Perry Cook, with its usage in a paper titled “Musical Genre Classification of Audio Signals”. The purpose of the paper was to begin the process of automatic music genre annotation through a song’s instrumentation, rhythmic structure, and harmonic content (Tzanetakis & Cook, 2002). While Tzanetakis' attempt was the first of its kind, it has since exploded over the years accumulating over sixteen-hundred citations as other researchers have engaged with both the dataset and the initial research. For most as it relates to neural networks, most researchers have opted to use either a CNN, RNN or convolutional recurrent neural network (CRNN) in their experiments on GTZAN. In a comparative analysis between those three networks, their modified CNN achieved a 93.9% classification accuracy on GTZAN, their RNN 96% and their CRNN struggled achieving only 64.7% accuracy (Alam et al., 2023). A similar study between CNN and RNNs found them to perform at 83.7% accuracy and 74.1% accuracy respectively (Alshurideh et al., 2021).

When comparing the results of these two studies it becomes clear that the capabilities of each depend entirely on the implementation of the model. In general, RNNs are much more effective when it comes to handling sub features of data and how their summarization, while CNNs and their convolutions work off weighted averages and subsampling (Choi et al., 2016). Since each have their own strengths, they are commonly combined into a CRNN and can achieve remarkable results when properly implemented with the RNN handling temporal pattern aggregation and the CNN used for local feature extraction of inputs (Choi et al., 2016). Due to limitations we chose to forgo using a CRNN and instead compared a RNN and CNN with a FCNN as a baseline. A feed-forward network which was trained on data similar to GTZAN in a set known as AMG1608 was able to achieve an accuracy of 77.6% (Imran et al., 2017) which is partly why we chose to use our own FCNN as the baseline model.

While automated music genre classification began with Tzanetakis in 2002, the introduction of large-scale streaming and broadcasting services such as Spotify and Apple Music commercially implemented it for data mining and music recommendations for users (Aydin & Elbir, 2020). Large scale implementations of these systems as part of a business model have created value in further research and refinement of networks which can classify music. Genre is typically the center of how users organize music (Edward et al., 2024), and capitalization of user genre preferences leads to consumer retention which in turn generates revenue for companies (Aydin & Elbir, 2020). Since streaming services have access to such an extensive catalog of songs, automated classification allows for them to provide personalized recommendation to users based off of listening history (Aydin & Elbir, 2020). These practical applications of music genre classifications shed some insight into the importance of genre classification systems as it relates to the music industry.

**3. Methodology**

**3.1 Dataset**

The GTZAN dataset consists of various representations of songs from the following ten genres: blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. The dataset was collected in the years 2000 and 2001, although it should be noted that the specific dates that each song was composed and produced were not included in the dataset. Song data was derived from various audio mediums including cd, radio, and direct microphone recordings in order to accurately represent a variety of recording conditions.

The dataset itself is composed of three different representations of music data. The first of these representations is a collection of 100 waveform files from each of the ten genres. This data is stored in the “genres\_original” directory. Features from these files can be extracted and turned into tensors using the torchaudio package.

The second of these representations is a collection of 1000 mel spectrograms, one image for each waveform file. These visual representations of each song’s frequency spectrum and their amplitude over time provide the ability to accurately extract features of each song using a convolutional neural network like other image classification models. This data is stored in the “images\_original” directory.

The third of these representations consists of two csv files containing extracted features of each song. The first file named “features\_30\_sec.csv” contains mean and variance calculations for the root mean square, spectral centroid, spectral bandwidth, spectral roll off values, zero crossing rate, harmony, and twenty Mel-Frequency Cepstral Coefficients. The data also includes a tempo for each song and its true genre label. The second file named “features\_3\_sec.csv” contains the same feature calculations as the previous file, however these calculations were performed over three second segments of each thirty second song generating ten times the amount of data that can be used to train a given model. The complete dataset can be downloaded [here](https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification/data).

**3.2 Implementation of Fully Connected Feed Forward Network**

Three feed forward networks were created for the three and thirty second datasets, each with their own unique network architecture: one with 8 hidden layers of width 5, one with 16 hidden layers of width 10, and one with 24 hidden layers of width 15. These networks were created with arbitrary dimensions; however, it was hypothesized that the variance in the dimensions of each network would provide some level of understanding of the relative scale of each network required for effective genre classification. In addition, each network used stochastic gradient decent optimization and cross entropy loss for loss calculation.

Each network was trained over 1000 epochs on two thirds of their respective dataset after it was randomly shuffled. The other third of the data was used for assessing the accuracy of each network. The three and thirty second networks which generalized the best were then saved to the directory labeled “Best\_Performing\_Networks” for future use.

Through testing it was determined that the best performing network contained 16 layers of width 10 for both the 30 second and 3 second datasets. The network trained on the 3 second dataset achieved an accuracy of approximately 25.526%, while the network trained on the 30 second dataset achieved an accuracy of approximately 12.575%; this disparity between the two networks is not unexpected, as the 3 second dataset is ten times as large as the 30 second dataset and the amount of data models are trained on can often be a solid predictor of their related success rates.

This accuracy could be improved by implementing techniques like those utilized by Imran et. Al in their feed forward network mentioned in section 2. Their implementation involved utilizing a feature selection algorithm to ignore features which do not appear to have an effect on classification decision. This pruning process decreased their feature set from 11 to 6. Furthermore, their network’s architecture and relevant hyperparameters were most likely tuned to a greater degree and the feedforward networks in this project; however, a peak accuracy of 25% is more accurate than the expected 10% accuracy achieve by simply guessing, and provides enough evidence to warrant using feed forward networks as the basis for the other networks created in this experiment.

**3.3 Implementation of a Convolutional Neural Network**

Designing the Convolutional Neural Network took a bit of time. Although only one was used, the internal design of it went through different iterations before one was settled on. This was a matter of making changes to get it to function properly. From the results presented in the research papers, a CNN should be able to identify the genres with a high level of accuracy with the use of the image data. We hoped that our attempt would be able to find success.

To begin our attempt, we needed to have training and testing images. The GTZAN dataset had a folder of images called images\_original which contained subfolders of different genres. Each of these folders contained one hundred mel spectrogram images that were 432 x 288. The decision was made to have 80 images for each genre in the training set and 20 images in the test set. This was manually done prior. ImageFolder() was used to have the images be prepared in a way that they were classified by genres.

For the CNN itself, there are three 2D convolution layers for each, a kernel size of 3, a stride of 1, and a padding size of 1 is used. The in-size of the first convolution layer is 3 because of RGB. The size goes to 32. The in-size of the second is 32 and the out-size is double at 64. The next layer goes from 64 to 128. Between each layer a pooling layer is created. Next a linear layer goes from 128 \* 54 \* 36 to 512. The numbers 54 and 36 are used to account for the pooling factor of two in the three pooling layers. (432 and 288 were divided by 2 three times). The final layer goes from 512 to 10(outsize) which is the number of unique genres in the dataset. The forward function takes the convolution layers with x, and this is passed through a ReLu activation function. This is applied in max pooling to down sample. This is done for each three layers and x is updated each time.

Running the CNN was time consuming. This contributed to the decision of making the number of epochs settled on was 25. Each epoch took roughly a minute to run. The reason for this was an epoch of 10 produced a training accuracy of 42%. An attempt with 100 epochs was attempted but after one hour and at epoch 57, it was determined to not continue. It is worth mentioning however that the accuracy for the training at that point was 99.65%, this is a very high accuracy. The reason for not continuing with this was because of how long it was taking and the high risk of the model being overfitted by the end. It was noted from that attempt that 90% accuracy was reached around epoch 20. That is the reason 25 was selected.

The results of the CNN were promising and with each epoch the accuracy of the training data increased. The final training accuracy was 99.62%. When evaluated against the test set, only 37% accuracy was reached. This could have to do with the structure of the CNN and better hyperparameters would need to be tested. It could be that the model was overfitted on the training data. Another issue could be with how the images were divided into training and testing groups. It might be that more images need to be trained on or there is not enough to test against. There could be many changes made that could produce better results.

The accuracy that resulted from the CNN in training was impressive, but the test accuracy was not that great. In terms of what the papers said, what we found lined up in that the CNN performed better than other methods. Alterations to the CNN could be attempted with changes to hyperparameters but because of how long it takes to run and the timeframe of the project, we opted not to investigate further. Although accuracy could have been improved upon, the results we had were satisfying.

**3.4 Implementation of a Recurrent Neural Network**

For the RNN we used four networks; RNN-1a and RNN-1b which consisted of a single LTSM followed by a fully connected layer with 64 hidden layers which was then followed by a dropout layer with a rate of .5 to help with generalization, they were each trained on three-second and thirty-second data, respectively. The second set of networks, RNN-2a and RNN-2b, consisted of two LTSMs setup as a bidirectional LTSM, with the same setup for the fully connected and dropout layer. All four models were trained on 1000 epochs and the numerical data from GTZAN dataset as opposed to the mel spectrogram images the CNN was trained on.

As for performance, RNN-2a/b achieved the greatest accuracy, but they were not far off from what RNN-1a/b achieved. In the end RNN-1a reached 14.2% accuracy, RNN-1b reached 14.4% accuracy, RNN-2a reached 16.75% accuracy and RNN-2b achieved 19.4% accuracy. When compared to the much higher ranges found in current literature surrounding RNNs for genre classification this was unexpected. Upon further review it was determined that our RNNs could have been improved with the addition of another fully connected layer separated by a ReLu activation function, though it is worth noting that the other RNNs were trained mostly on mel spectrogram images (Eshwarappa et al., 2022).

**4. Conclusion and Future Work**

Through experimentation, it can be determined that the best structure out of the one’s mentioned is a CNN which analyzes mel spectrograms of song data. Allowing networks of this architecture more epochs to learn as well as a larger dataset and more tuning could provide a higher accuracy score than the model’s generated in this project. In addition to these changes, experimentation with ensembles of networks, or even combinations of recurrent convolutional neural networks may improve results.

While this research project did provide compelling evidence for the effectiveness of each of the various network structures discussed, the most effective improvement that could be made to turn these networks into a tool that can be used is to create a more general interface for processing and classifying song data. This could involve more robust data processing that would allow for inputs of diverse sizes, utilization of various music provider API services to collect larger datasets, and even a graphical interface or custom API so that these networks could be implemented in other software solutions in the future.

References

Alam, S., Noman, M., Prodhan, S. Z., & Rafi, Q. G. (2023). Comparative Analysis of Three Improved Deep Learning Architectures for Music Genre Classifications. *International Journal of Information Technology and Computer Science*(2), 1-14. doi:10.5815/ijitcs.2021.02.01

Alshurideh, M., Hejazi, H. D., Khamees, A. A., & Salloum, S. A. (2021). Classifying Audio Music Genres Using CNN and RNN. *International Conference on Advanced Machine Learning Technologies and Applications*, (pp. 315-323). doi:10.1007/978-3-030-69717-4\_31

Aydin, N., & Elbir, A. (2020). Music genre classification and music recommendation by using deep learning. *56*, pp. 627-629. Electronic Letters. doi:https://doi.org/10.1049/el.2019.4202

Choi, K., Fazekas, G., Sandler, M., & Cho, K. (2016). *Convolutional Recurrent Neural Networks for Music Classification.* London, UK: Queen Mary University of London.

Edward, A. F., McDonald, C., & Rafferty, P. (2024). Playlists and genre: the role of music genre in Spotify’s playlists. *Journal of Documentation*. doi:https://doi.org/10.1108/JD-08-2023-0152

Eshwarappa, V., Kakarla, C., Oghaz, M. M., & Saheer, L. B. (2022). Recurrent Neural Networks for Music Genre Classification. *International Conference on Artificial Intelligence*, (pp. 267-279). doi:10.1007/978-3-031-21441-7\_1

Imran, D., Radi, M., Tahir, M. A., & Wadiwala, H. (2017). Semantic Feature Extraction using Feed-Forward Neural Network for Music Genre Classification. *Asian Journal of Engineering, Sciences & Technology, 2*(1).

Tzanetakis, G., & Cook, P. (2002). Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing.* *10*, pp. 293-302. IEEE. doi:10.1109/TSA.2002.800560