Innovating Composite Mood Genre Classification in Music Through Convolutional Neural Networks

Project Final Report and Presentation

Zongru Sun
DSBA M2
CentraleSupelec
Paris, France
zongru.sun@student-cs.fr

Zhongke Sun
DSBA M2
CentraleSupelec
Paris, France
zhongke.sun@student-cs.fr

Wenxi Huang
DSBA M2
CentraleSupelec
Paris, France
wenxi.huang@student-cs.fr

1 Abstract

This report presents a novel approach in music classification, focusing on Composite Mood Genre (CMG) classification using Convolutional Neural Networks (CNNs). Addressing the limitation of current music classification systems in capturing the multifaceted nature of human emotions, this study innovates by categorizing songs into CMGs. The methodology incorporates advanced CNN architectures, processing a diverse dataset to enhance the accuracy and depth of music classification based on emotional content.

Key findings demonstrate that our CNN models, specifically CNN-64, significantly outperform traditional methods, particularly in genres like metal, jazz, and blues, in CMG like “Insecure But Inessential” and “Relax But Energetic”. However, further data and training are required for genres like disco and reggae. This advancement in music classification is not only academically significant but also holds potential for improving music recommendation systems, offering users a more tailored listening experience based on complex emotional states.

The implications of this research extend beyond music classification, contributing to the broader understanding of the interaction between music and human emotion. Our approach sets a new direction for future research in music information retrieval and opens avenues for more emotionally intelligent music recommendation systems.

2 Introduction and Motivation

In the realm of modern music streaming platforms like Spotify and Apple Music, a significant issue persists in the classification of diverse musical genres - the existing method of classifying music based on singular mood tags, such as happy or sad, fails to capture the complexity of human emotions. Human moods are multifaceted and often contradictory, yet the current systems lack the nuance to represent these intricate emotional compositions accurately.

Our proposal aims to address these shortcomings by enhancing the capabilities of audio feature-learning algorithms. Specifically, the primary objective is to refine:

The genres: Maximize the accuracy of music classification into genres.

The moods: Explore the moods within music and to categorize songs into composite mood genres. This advancement will pave the way for a more refined music recommendation system. Imagine a scenario where users can select a combination of mood keywords “Negative but Hopeful”. This customizable approach will enable users to fine-tune their mood preferences and receive tailored music recommendations based on their nuanced emotional states.

The importance of addressing this problem extends far beyond the realm of music classification and recommendation. Music serves as a profound repository of human sentiment and reflection. By accurately identifying the composite moods in music, we not only enrich the overall music listening experience for individuals, but also preserve human’s ability in identifying, understanding, and appreciating their complex but real moods. Besides, for business application, the ability to offer music that resonates with the complex and varied moods of listeners will greatly enhance their emotional connection to the music, thus augmenting their overall satisfaction with the streaming platforms.

3 Problem Definition

The challenge lies in developing a CNN model that can effectively discern and categorize the nuanced emotional content of music tracks into Composite Mood Genres (CMG). Related formulars are discussed in Methodology Part. Traditional genre classification systems often overlook the intricate emotional layers in music. Our approach seeks to fill this gap by leveraging the pattern recognition capabilities of CNNs to interpret complex emotional cues in music, thereby innovating the field of music genre classification.

Constraints: All the machine learning tool used is in the filed of supervised learning. Limited dataset is acquired from GTZAN.

Optimize: The accuracy of our CNN's test prediction and the Performance illustrated from HeatMap.

4 Related Work

The field of music classification, particularly in the context of mood and emotion, has been a subject of extensive research. Our project builds upon this foundation, integrating and innovating upon existing methodologies.

State-of-the-Art Techniques: Recent studies have increasingly utilized deep learning methods, especially CNNs, for music genre classification. For instance, Choi et al. (2017) demonstrated the effectiveness of CNNs in music genre classification, emphasizing their superiority over traditional methods in handling complex datasets. Our project extends this approach to specifically focus on mood genre classification.

Mood-Based Music Classification: Existing research in mood-based classification, like Yang and Chen (2012), typically employs simpler machine learning models. Our project differs by using advanced CNN architectures, aiming for a more nuanced understanding of musical moods.

Emotion and Music: The work of Juslin & Västfjäll (2008) on the psychological framework of music and emotion provides a theoretical basis for our approach. We leverage their insights into emotional responses to music to inform our classification categories.

Innovation in Classification Schemes: Unlike traditional genre classification, which often limits a piece of music to a single genre, our project aims to recognize the composite nature of musical moods, acknowledging that a single track can encompass multiple emotional dimensions.

Each of these references informs our approach, with our project aiming to bridge the gap between traditional genre classification and the more complex, emotion-driven categorizations that modern deep learning techniques can facilitate.

5 Methodology

This section will introduce our methodology to **pre-process data and classify music using Convolutional Neural Networks**. While numerous studies exist in the domain of music classification and sentiment analysis, the proposed approach integrates **Convolutional Neural Networks (CNNs)** to advance the accuracy and depth of audio feature learning. Through this approach, we seek to revolutionize the way music is perceived, categorized, and recommended on leading streaming platforms. Here are main steps we took to implement our approach:

5.1 Dataset Choosing

We use GTZAN Genre Collection as our dataset. This dataset was used for the well-known paper in genre classification "Musical genre classification of audio signals" by G. Tzanetakis and P. Cook in IEEE Transactions on Audio and Speech Processing 2002.

5.2 Dataset Setup

This GTZAN Genre Collection will be used for two classifications-music genre classification and composite mood classification.

Therefore, for the first task, we have ten different music genres, they are 'metal', 'pop', 'disco', 'classical', 'country', 'hiphop', 'blues', 'rock', 'reggae', 'jazz'. Each files folder has 100 .au audio files and 1000 .au audio files total.

And for the composite mood classification, we really spend much time to do marketing research. We asked our friends to help us mark the composite genre of some of songs in the dataset. Finally we conclude ten different categories: 'NegativeButHopeful', 'RelaxButEnergetic', 'TotallyPositive', 'TotallyNegative', 'PositiveButVoid', 'CalmButEpic', 'InsecureButInessential', 'ConcernedButLight', 'CalmButWarm', and 'Other'.

5.3 Spectrogram Generation and Train Data Split

Each audio will be transformed to a spectrogram which is a visual representation of spectrum of frequencies over time. The pixels and locations on the graph will replace the traditional time axis and lines. This is why we can utilize convolutional neural networks to analyze the features of audio data.

In the data split, we split 80% of data for training, 10% of data for validation and 10% for testing.

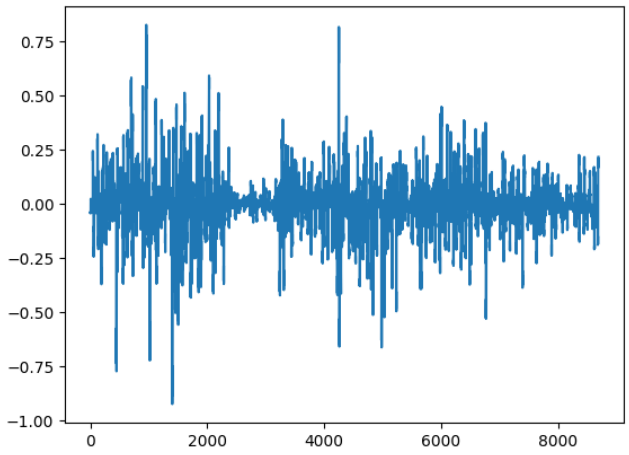

Figure 1: Spectrogram Graph Generated By Audios.

5.4 Convolutional Neural Networks (CNNs)

5.4.1 Basic Introduction

Convolutional neural network (CNN) is a regularized type of feed-forward neural network that learns feature engineering by itself via filters (or kernel) optimization. Vanishing gradients and exploding gradients, seen during backpropagation in earlier neural networks, are prevented by using regularized weights over fewer connections.

CNNs are also known as Shift Invariant or Space Invariant Artificial Neural Networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation-equivariant responses known as feature maps.

Feed-forward neural networks are usually fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) Robust datasets also increase the probability that CNNs will learn the generalized principles that characterize a given dataset rather than the biases of a poorly populated set.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns to optimize the filters (or kernels) through automated learning, whereas in traditional algorithms these filters are hand-engineered. This independence from prior knowledge and human intervention in feature extraction is a major advantage.

5.4.2 Overall architecture

CNNs are comprised of three types of layers. These are convolutional layers, pooling layers, and fully connected layers. When these layers are stacked, a CNN architecture has been formed.

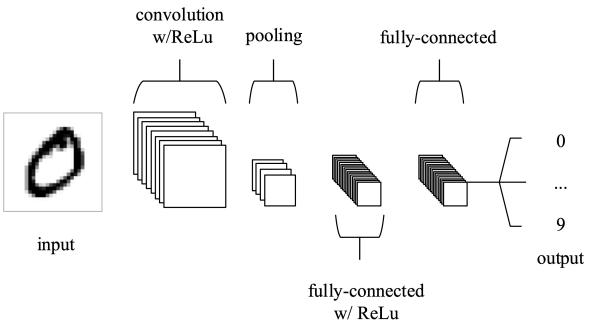

Figure 2: A simple CNN architecture, comprised of just five layers

5.4.3 Neurons

Neural networks are comprised of numerous interconnected neurons. Each neuron receives a linear combination of inputs and is initially simply weighted linearly, and later a nonlinear activation function is added to each neuron, resulting in a nonlinearly transformed output. The connections between any two neurons represent weighted values known as weights. Different weights and activation functions result in different outputs from the neural network.

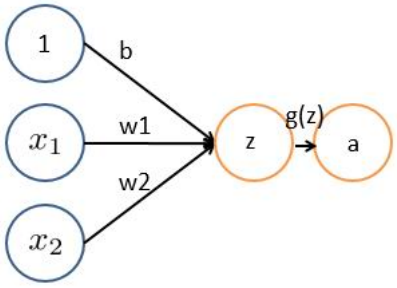

Figure 3: The Neurons

The structure of the neuron is based on the form $wx + b$, where:

- x_1, x_2 represents output vector.
- w_1, w_2 are weights, each input is assigned a weight.
- b is bias.
- $g(z)$ is activation function.
- a is output.

5.4.4 Activation Function

Activation function of a node in an artificial neural network is a function that calculates the output of the node (based on its inputs and the weights on individual inputs). Modern activation functions include the smooth version of the ReLU, the GELU, which was used in the 2018 BERT model, the logistic (sigmoid) function used in the 2012 speech recognition model developed by Hinton et al, the ReLU used in the 2012 AlexNet computer vision model and in the 2015 ResNet model.

Sigmoid Function

$$g(z) = \frac{1}{1 + e^{-z}}$$

Where z is a linear combination, like $z = b + w_1 \times x_1 + w_2 \times x_2$. By substituting an enough large positive number or an enough small negative number into the $g(z)$, the result converges to 0 or 1. In other words, the Sigmoid function is functionally equivalent to compressing a real number between 0 and 1. When z is a very large positive, $g(z)$ converges to 1, similarly, when z is a very small negative number, $g(z)$ converges to 0.

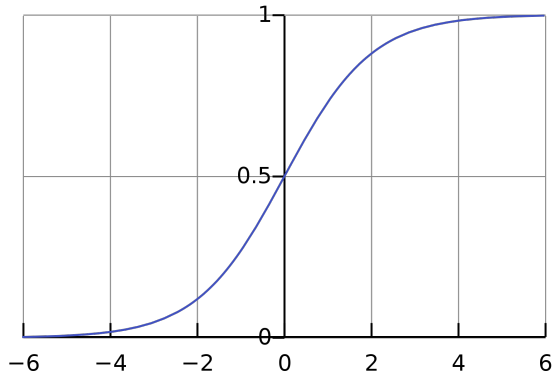

Figure 4: Sigmoid Function Graph

Rectified Linear Unit (ReLU) Function

$$\sigma(x) = \begin{cases} \max(0, x), x \geq 0 \\ 0 \quad , x < 0 \end{cases}$$

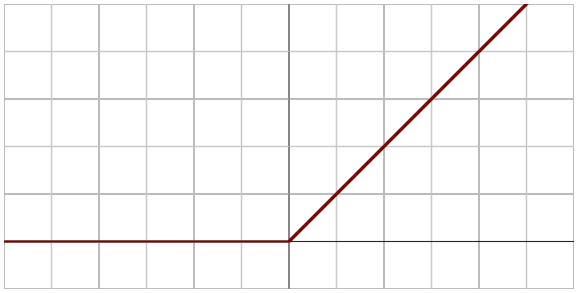

Figure 5: Rectified Linear Unit (ReLU) Function Graph

Softmax Function

Softmax is an activation function that scales numbers/logits into probabilities. The output of a Softmax is a vector with probabilities of each possible outcome. The probabilities in vector sums to one for all possible outcomes or classes.

Mathematically, Softmax is defined as,

$$S(y)_i = \frac{\exp(y_i)}{\sum_{j=1}^{n} \exp(y_i)}$$

Where:

y: is an input vector to a Softmax function. It consists of n element for n classes (possible outcomes).

y_i: the $i-th$ element of the input vector. It can take any value between -inf and +inf.

$\exp(y_i)$: standard exponential function applied on y_i.

$\sum_{j=1}^{n} \exp(y_j)$: A normalization term. It ensures that the values of output vector $S(y)_i$ sums to 1 for $i-th$ class and each of them and each of them is in the range 0 and 1 which makes up a valid probability distribution.

n: Number of classes (possible outcomes).

5.4.5 Convolutional Layer

The convolutional layer can generate a set of parallel feature maps by sliding different convolutional kernels over the input image and performing specific operations. Additionally, at each sliding position, the convolutional kernel and the input image undergo an element-wise multiplication and summation operation to project the information within the receptive field to an element in the feature map. This sliding process is referred to the stride Z_s, which Z_s serves as a factor controlling the output feature map size. The size of the convolutional kernel is considerably smaller than that of the input image, and it acts in an overlapping or parallel manner on the input image. All elements in a feature map are computed using a single convolutional kernel, implying that a feature map shares the same set of weights and bias terms.

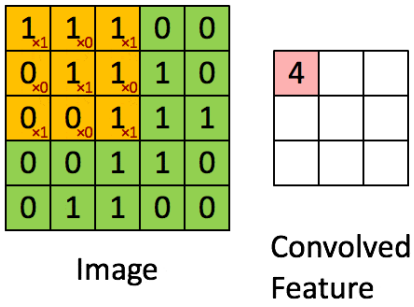

Figure 6: The Computation of Convolutional Layer

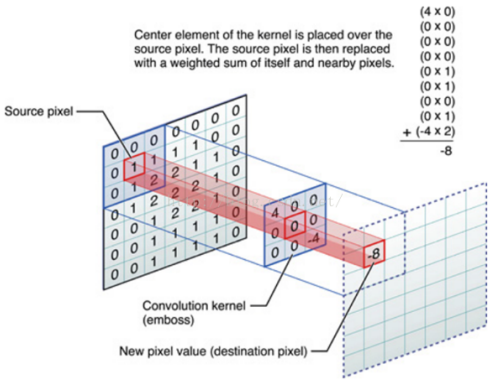

Figure 7: Projective Computation of Convolutional Kernels

5.4.6 Pooling Layer

Pooling layers aim to gradually reduce the dimensionality of the representation, and thus further reduce the number of parameters and the computational complexity of the model.

The pooling layer operates over each activation map in the input and scales its dimensionality using the "MAX" function. In most CNNs, these come in the form of max-pooling layers with kernels

of a dimensionality of 2 x 2 applied with a stride of 2 along the spatial dimensions of the input.

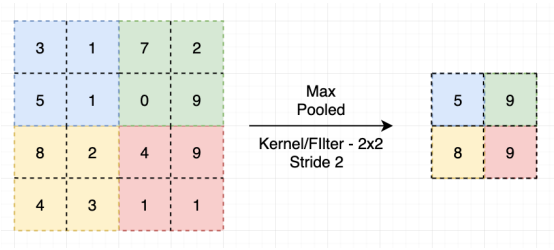

Figure 8: Pooling Layer

Intuitively, the effectiveness of this mechanism lies in the fact that the precise location of a future is far less crucial than its relative position to other features. The pooling layer continuously reduces the spatial size of the data, leading to a decrease in the number of parameters and computational load, thereby helping to control overfitting to some extent. Typically, pooling layers are periodically inserted between convolution layers in the architecture of a Convolutional Neural Network (CNN). Pooling operations provide another form of translational invariance. As convolutional kernels serve as feature detectors, various edges in an image can be easily identified through convolutional layers. However, the features discovered by convolutional layers often exhibit excessive precision. Even when capturing rapid successive shots of an object, the pixel positions of the object's edges in the photos are unlikely to be completely consistent. The pooling layer helps mitigate the sensitivity of convolutional layers to such edge details.

The pooling layer computes the output within a pooling window (depth slice) and then shifts the pooling window based on the specified stride. The illustration below depicts the most widely used pooling layer, a 2D max pooling layer with a stride of 2 and a pooling window of 2×2. At intervals of two elements, 2×2 blocks are delineated from the image, and the maximum value is taken for each of the four numbers within each block. This process results in a reduction of 75% in the data volume.

$$f_{X,Y}(S) = \max_{a,b=0} S_{2X+a,2Y+b}$$

5.4.7 Fully connected layer

The fully connected layer contains neurons of which are directly connected to the neurons in the two adjacent layers, without being connected to any layers within them.

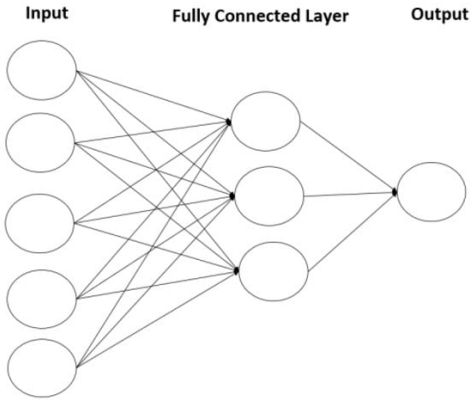

Figure 19: Fully Connected Layer

5.5 CNN Structures

5.5.1 LeNet-5

LeNet is a convolutional neural network structure proposed by LeCun et al. in 1998. As a representative of the early convolutional neural network, LeNet possesses the basic units of convolutional neural network, such as convolutional layer, pooling layer, and full connection layer, laying a foundation for the future development of convolutional neural network. As shown in the figure (input image data with 32*32 pixels): LeNet-5 consists of seven layers. In addition to input, every other layer can train parameters. Cx represents convolution layer, Sx represents subsampling layer, Fx represents complete connection layer, and x represents layer index.

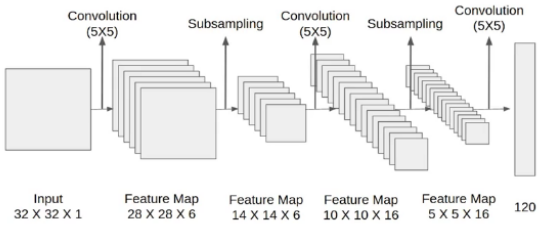

Figure 10: Structure of LeNet-5

Layer C1 is a convolution layer with six convolution kernels of 5 x 5 and the size of feature mapping is 28 x 28, which can prevent the information of the input image from falling out of the boundary of convolution kernel.

Layer S2 is the subsampling/pooling layer that outputs 6 feature graphs of size 14x14. Each cell in each feature map is connected to 2x2 neighborhoods in the corresponding feature map in C1.

Layer C3 is a convolution layer with 16 5x5 convolution kernels. The input of the first six C3 feature maps is each continuous subset of the three feature maps in S2, the input of the next six feature maps comes from the input of the four continuous subsets, and the input of the next three feature maps comes from the four

discontinuous subsets. Finally, the input for the last feature graph comes from all feature graphs of S2.

Layer S4 is similar to S2, with size of 2x2 and output of 16 5x5 feature graphs.

Layer C5 is a convolution layer with 120 convolution kernels of size 5x5. Each cell is connected to the 5x5 neighborhood on all 16 feature graphs of S4. Here, since the feature graph size of S4 is also 5x5, the output size of C5 is 1x1. So S4 and C5 are completely connected. C5 is labeled as a convolutional layer instead of a fully connected layer, because if LeNet-5 input becomes larger and its structure remains unchanged, its output size will be greater than 1x1, i.e. not a fully connected layer.

F6 layer is fully connected to C5, and 84 feature graphs are output.

5.5.2 CNN-64 (DCNN)

CNN-64 is a deep convolutional neural network that, in comparison to traditional CNNs (convolutional neural networks), exhibits an increased depth in its network architecture, featuring a greater number of convolutional layers and parameters. This augmented depth allows the network to learn more abstract and advanced feature representations. Simultaneously, by enhancing the network depth, DCNN-64 achieves a larger receptive field, implying a more extensive input region for each neuron. This facilitates the network's comprehension of a broader contextual range, particularly advantageous for complex image-related tasks.

Typically, global average pooling is employed in CNN-64, replacing fully connected layers to reduce the number of parameters and mitigate the risk of overfitting. This aids in improving the network's generalization performance.

In the context of CNN-64, 64 classifiers are utilized, and the network structure is illustrated in the accompanying diagram:

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 128, 431, 1)]     0

 conv2d_5 (Conv2D)           (None, 126, 429, 64)      640

 max_pooling2d_3 (MaxPoolin  (None, 63, 214, 64)       0
 g2D)

 dropout_8 (Dropout)         (None, 63, 214, 64)       0

 conv2d_6 (Conv2D)           (None, 61, 212, 64)       36928

 max_pooling2d_4 (MaxPoolin  (None, 30, 106, 64)       0
 g2D)

 dropout_9 (Dropout)         (None, 30, 106, 64)       0

 conv2d_7 (Conv2D)           (None, 28, 104, 64)       36928

 max_pooling2d_5 (MaxPoolin  (None, 14, 52, 64)        0
 g2D)

 dropout_10 (Dropout)        (None, 14, 52, 64)        0

 conv2d_8 (Conv2D)           (None, 12, 50, 64)        36928

 max_pooling2d_6 (MaxPoolin  (None, 6, 25, 64)         0
 g2D)

 dropout_11 (Dropout)        (None, 6, 25, 64)         0

 flatten_2 (Flatten)         (None, 9600)              0

 dense_5 (Dense)             (None, 10)                96010

=================================================================
Total params: 207434 (810.29 KB)
Trainable params: 207434 (810.29 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 112: Structure of CNN-64

6 Evaluation

6.1 Result of the Music Genre Classification

We will perform cross-validation on the dataset, splitting it into training and testing sets multiple times to check for consistency in performance. The metrics to evaluate our models mainly include:

Accuracy: it quantifies the percentage of accurately classified test samples.

Heatmap: it is visual representations that use color gradients to show the intensity or concentration of values in a two-dimensional space. In the context of deep learning, heatmaps are often used to visualize the importance or activation of certain regions in an input, especially in tasks like object detection, image segmentation, or attention mechanisms.

We can conclude that our trained Convolutional Neural Network model has a 0.8241 accuracy. And heatmap demonstrates that our LeNet-5 model perform well in detection of metal, jazz, and blues music, but we still need more data to train the model to make it perform better in detecting disco music.

```
[ ] lenet.evaluate(x=x_test, y=y_test)

    28/28 [==============================] - 0s 7ms/step - loss: 0.7717 - accuracy: 0.8241
    [0.7716569304466248, 0.8241379261016846]
```

Figure 12: Accuracy of LeNet-5 in Music Genre Classification

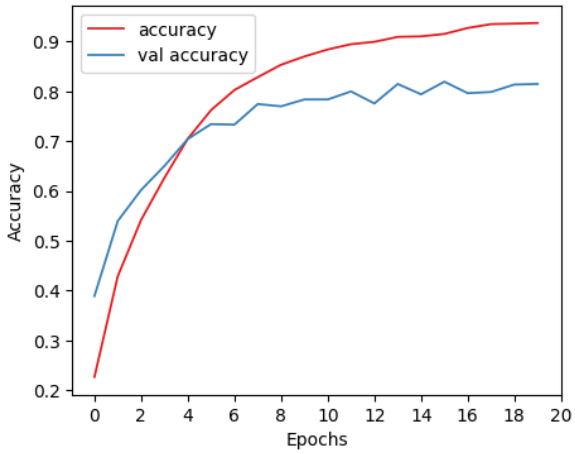

Figure 13: Accuracy Comparison

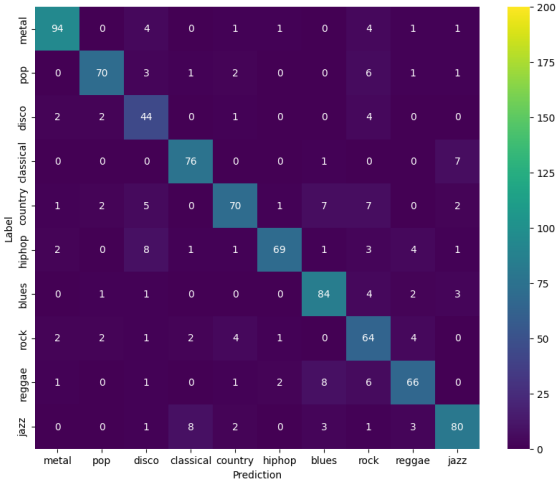

Figure 14: Heatmap of LeNet-5 Model in Music Genre Classification

And for the CNN-64, we can find our model have a better performance, it has a higher accuracy and can detect more music according to heatmap of CNN-64.

```
[ ] model.evaluate(x=x_test, y=y_test)

    28/28 [==============================] - 1s 18ms/step - loss: 0.9293 - accuracy: 0.8517
    [0.9292983412742615, 0.8517241477966309]
```

Figure 15: Accuracy of CNN- 64 in Music Genre Classification

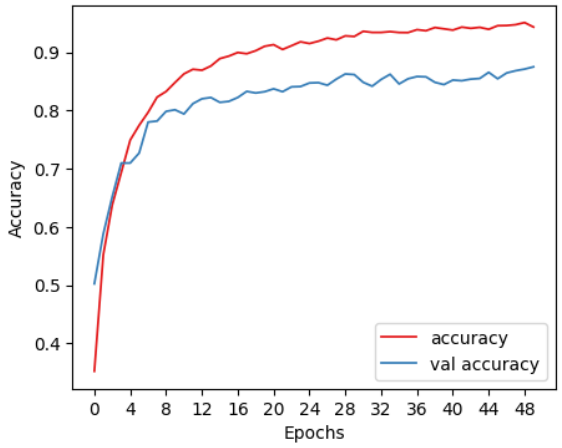

Figure 16: Accuracy Comparison

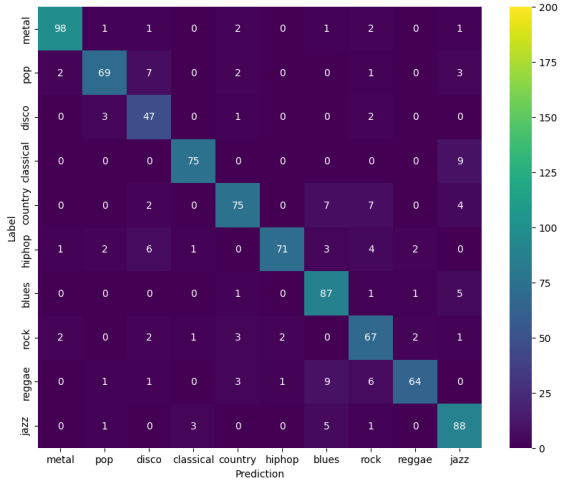

Figure 17: Heatmap of CNN-64 in Music Genre Classification

6.2 Result of the Composite Mood Classification

In the task of composite mood classification, LeNet-5 model has a 0.5911 accuracy and CNN-64 has a 0.6644 accuracy. Similar to music genre classification, CNN-64 performs better in accuracy.

```
29/29 [==============================] - 0s 5ms/step - loss: 1.4980 - accuracy: 0.5911

[1.4980307817459106, 0.5911111235618591]
```

Figure 18: Accuracy of LeNet-5 in Composite Mood Classification

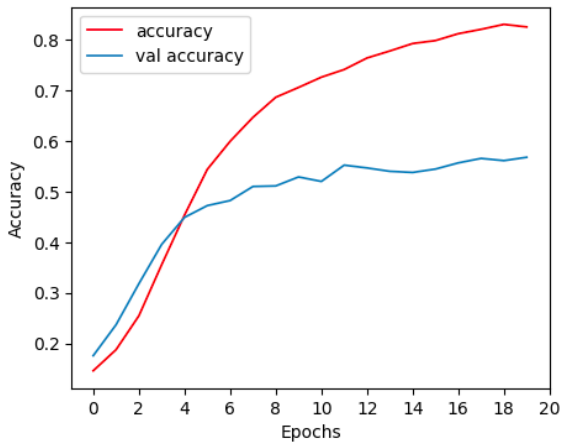

Figure 19 Accuracy Comparison

```
29/29 [==============================] - 0s 7ms/step - loss: 2.0630 - accuracy: 0.6644
[2.0630264282226562, 0.6644444465637207]
```

Figure 19: Accuracy of CNN-64 in Composite Mood Classification

According to heatmap, we found our model perform well in the 'InsecureButInessential' and 'RelaxButEnergetic' music. Specially, we need to do more train and have more train data in 'CalmButWarm' and 'TotallyNegative'.

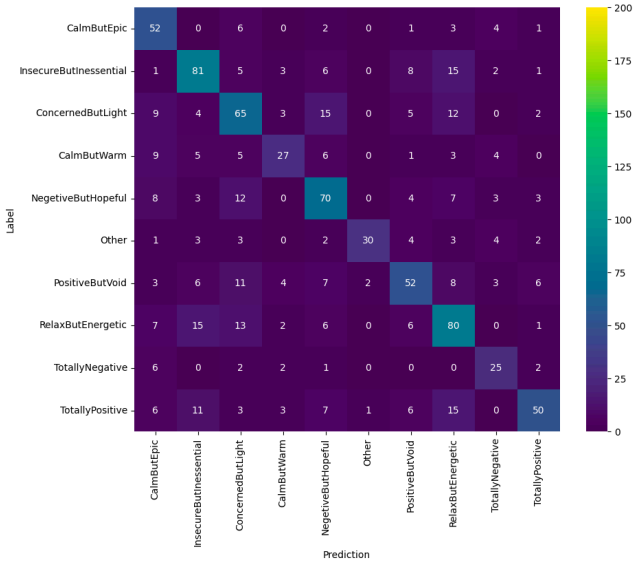

Figure 20: Heatmap of LeNet-5 in Composite Mood Classification

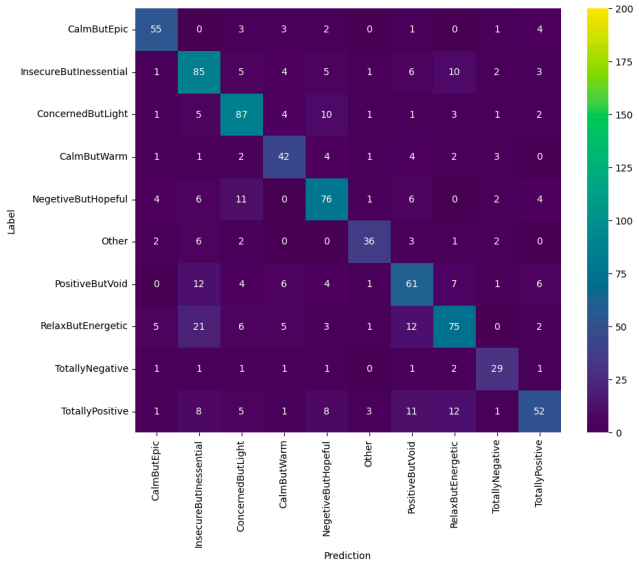

Figure 21: Heatmap of CNN-64 in Composite Mood Classification

7 Conclusion

In the task of Music Genre Classification, CNN-64 performs better than LeNet-5. And I must highlight that in specific music classification, these two models perform very well (in metal, jazz, blues). In the future, we need to train specific kind of music such as disco, reggae, etc.

Furthermore, in the composite mood classification, due to our sufficient music dataset with 'InsecureButLight' and 'ConnernedButLight' mood, our model has great ability to detect the mood. And at the same time, we also need to collect more dataset with 'TotallyNegative' m

For further researcher, they can explore more datasets and longer music to analysis as our 30s audio may not represent a piece of music completely. And we can also deduce from the distribution of mood generes that classical music contains more mood information than others, such as Raggae and Hiphop, as they are developed based on the old genres – classical. In conclusion, this is a brave try to explore the balance between subjective and objective.

REFERENCES

[1] Michael I Mandel and Dan Ellis. 2005. *Song-level features and support vector machines for music classification.* New York.

[2] Tom LH Li, Antoni B Chan, and A Chun. 2010. *Automatic musical pattern feature extraction using convolutional neural network.* Hong Kong.

[3] S. Prince, J. J. Thomas, S. J. J, K. P. Priya and J. J. Daniel, *Music Genre Classification using Deep learning - A review*, 2022 6th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), Bangalore, India, 2022, pp. 1-5, DOI: 10.1109/CSITSS57437.2022.10026394.

[4] Xiaosong Jia, *A Music Emotion Classification Model Based on the Improved Convolutional Neural Network*, Computational Intelligence and Neuroscience, vol. 2022, Article ID 6749622, 11 pages, 2022. DOI:doi.org/10.1155/2022/6749622.

[5] Yi-Hsuan Yang and Homer H. Chen. 2012. *Machine Recognition of Music Emotion: A Review*. ACM Transactions on Intelligent Systems and Technology, Vol. 3, No. 3, Article 40.

[6] Keunwoo Choi, George Fazekas, Mark Sandler, Kyunghyun Cho. 2017. *Convolutional Recurrent Neural Net- works for Music Classification*. DOI:https://doi.org/10.48550/arXiv.1609.04243.

[7] Venkatesan, Ragav; Li, Baoxin (2017-10-23). *Convolutional Neural Networks in Visual Computing: A Concise Guide.* CRC Press. ISBN 978-1-351-65032-8. Archived from the original on 2023-10-16. Retrieved 2020-12-13.

[8] Balas, Valentina E.; Kumar, Raghvendra; Srivastava, Rajshree (2019-11-19). *Recent Trends and Advances in Artificial Intelligence and Internet of Things. Springer Nature.* ISBN 978-3-030-32644-9. Archived from the original on 2023-10-16. Retrieved 2020-12-13.

[9] O'Shea, K., & Nash, R. (2015). *An introduction to convolutional neural networks*. arXiv preprint arXiv:1511.08458.

[10] Patrik N. Juslin and Daniel Västfjäll. 2008. *Emotional Responses to Music: The Need to Consider Underlying Mechanisms.* Behavioral and Brain Sciences, Vol. 31, Issue 5, pp. 559-575.

[11] Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. (1998). *"Gradient-based learning applied to document recognition" (PDF)*. Proceedings of the IEEE. 86 (11): 2278–2324. doi:10.1109/5.726791. S2CID 14542261.

[12] LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; Jackel, L. D. (December 1989). *"Backpropagation Applied to Handwritten Zip Code Recognition".* Neural Computation. 1 (4): 541–551. doi:10.1162/neco.1989.1.4.541. ISSN 0899-7667. S2CID 41312633.

[13] Lecun, Yann (June 1989). *"Generalization and network design strategies" (PDF).* Technical Report CRG-TR-89-4. Department of Computer Science, University of Toronto.