A Comparative Study of Multilayer Perceptron and Support Vector Machines in Musical Genre Classification

Samantha Georgina Isaac Munoz (230057658)

Abstract

The aim of this paper is to compare and evaluate in a comprehensive analysis two Neural Computing models, Multilayer Perceptron (MLP) and Support Vector Machines (SVM), in a multidimensional classification task of music genres based on their characteristics. A baseline model was made for each to determine a starting point for improvement of the models. Subsequently, a model training process was carried out using cycles, GridSearch, and the implementation of Optuna. For the verification of the results, the accuracy, loss graphs for the test and training data, and classification report were used.

1. Introduction

Music forms an integral core in human life and society. The impact it brings goes further afield than simple amusement, serving as a major social catalyst, a bridge for cultural connection, and a vehicle for moral and emotional expression [1]. Music genres are, by far, of vital importance in the delimitation of both the production and consumption parameters within such a vast and ever-growing industry. The genres are no fixed categories but, rather, a changing, interacting, and diversifying categories that are spurred on by new technologies and the changing tides of cultural landscapes.

The objective of this paper is to conduct a comprehensive analysis of two advanced Neural Computing models and their efficacy in the classification of the music genre based on their musical features. These models are Multilayer Perceptron (MLP) and Support Vector Machines (SVM) which are implemented using the libraries *PyTorch* and *Scikit-learn* respectively. Both models possess unique strengths in the processing and classification of complex datasets, and their capability to capture and learn from the relationships and patterns within music data positions them as a probable ideal tool for this task.

1.1 Multilayer Perceptron (MLP)

The MLP model represents one of the most recognized and commonly used models in neural networks because it is highly structured and organised [2]. It has three layers of nodes: an input layer, one or more hidden layers, and an output layer. The defining characteristic of MLP is the unidirectional feedforward flow of data from the input layer through the various hidden layers to the output layer. Finally, non-linear activation functions are applied to the nodes or neurons present in these layers, in order to capture and train over complex, nonlinear relationships existing between input and output data.

This model learns via backpropagation, which is a supervised training technique that changes exactly the weight of the connections between the nodes. This adjustment is made to minimise the difference between the predicted outcomes and the actual outcomes. It is suited for solving

a wide variety of challenging tasks, such as classification, retrogression, and pattern recognition.

1.2 Support Vector Classifier (SVC)

An SVC model is part of the Support Vector Machines (SVM), which are supervised learning methods for classification, regression, and outlier detection. The aim of SVC is to find the hyperplane that can best divide the different data classes in the feature space. This model attempts to maximise the distance between the nearest points of the support vectors and the separation hyperplane [3]. The larger the margin, the better the classifier's generalisation to new data. It is also important to highlight that when it is not possible to linearly separate the data, SVC can employ the kernel method to transform the feature space into a dimension where linear separation is possible.

2. Dataset

The dataset used in this project was obtained from Kaggle and focuses on the classification of musical genres, employing numerical values to depict their musical characteristics [4]. It comprises a total of 50,005 entries, featuring 16 attributes, both numerical and textual, in addition to an attribute designated to specify the label corresponding to one of the 10 distinct classes of musical genres. Figure 1 illustrates the balance among these classes, showcasing an equitable distribution and obviating the need for data balancing techniques.

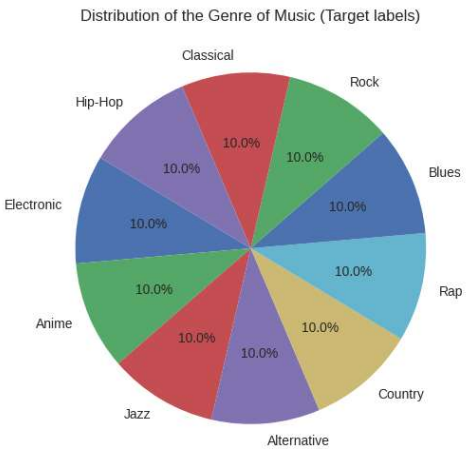

Figure 1. Distribution of the target labels per genre

Before using the data, it was important to perform a data cleansing procedure. Since the dataset was mostly error-free, the cleaning was minor. A few records were removed as they contained null or duplicate values; however, since they accounted for less than 1% of the total features, their removal wouldn't compromise the integrity of the information in future models.

A total of four features were discarded for being irrelevant to the project's goal. Furthermore, through the creation of a correlation heatmap, it was determined that these features had little significance in the classification of musical genres. During the exploration phase, it was noted that many data points were represented as text strings, even though they were numeric or floating-point values. Consequently, the data type for these entries was changed. Lastly, in the final step of the data cleaning process, for the '*tempo*' feature, several records displayed a '*?*' symbol, which was removed. Using the *fillna* method with the average value per genre, missing values in empty records were replaced with the genre-specific averages.

2.1. Insight of the Data

Following the data clean-up, a total of 13 features were identified for the classification and analysis of musical genres. In attempting to determine the presence of unique values for different characteristics by genre, it was found that these values can vary and are not strictly

constant within genre, which is anticipated. Nonetheless, some intriguing insights were discovered.

For instance, the boxplot in Figure 2 illustrates that the 'popularity' feature identifies that the most popular musical genres are Rap, Hip-Hop, and Rock. For some other features, it was concluded that there is no distinctive variation in some features between genres, such as with 'tempo', where most genres hover around 125 BPM, with Jazz and Classical being exceptions. Another interesting finding is with the feature 'energy', where most genres exhibit consistent values reflecting the energy in their songs, yet they also display variations across the range from 0 to 1, where 0 indicates low energy and 1 high energy. Classical music is noteworthy in this context, with an average energy level of 0.1, but it includes outliers above 0.5, indicating the presence of several energetic pieces, albeit less commonly.

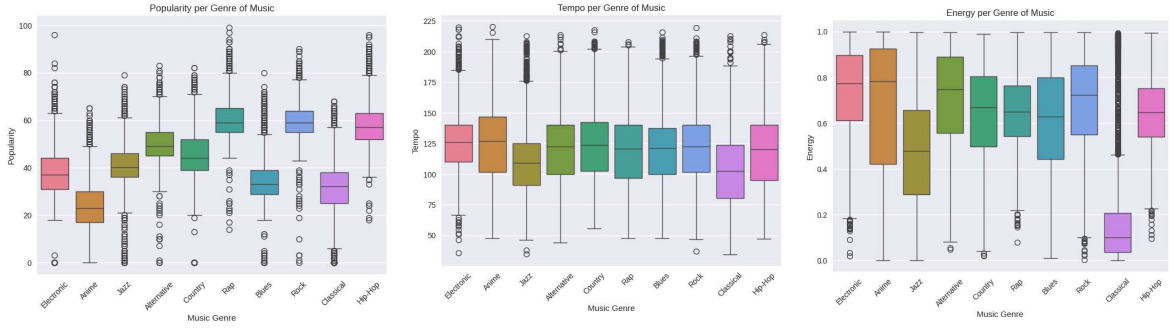

Figure 2. Three different boxplots to demonstrate the distribution of genres according to three different musical features: popularity, tempo and energy.

3. Models

3.1 Data Pre-processing

First the data was divided to a 70-30 ratio, training data and testing data respectively, while ensuring that the splitting is done randomly with the set seed for reproducibility. For the pipeline, there were two key transformations, numeric feature normalisation using StandardScaler and the categorical features being one-hot encoded with the use of *OneHotEncoder*. Encapsulated with a pipeline wrapper, these pre-process steps are done in such a way that it can be used for both training and testing sets, maintaining consistency.

3.2 Model Training and Validation

3.2.1 Multilayer Perceptron (MLP)

For the implementation of the MLP model with *Pytorch*, the target variables representing music genres were encoded into integer labels with *LabelEncoder*, and both features and labels were converted to *PyTorch* tensors to ensure compatibility with the framework's computational graph. These conversion standardised feature types to *float32* and label types to *long*. These tensors were then encapsulated within *TensorDataset* objects, facilitating efficient mini batch iteration during model training and evaluation through *DataLoader* instances.

A 3-layer architecture was applied: the size of the input layer was defined from the pre-processed feature set, and the number of neurons in two hidden layers containing 100 neurons

each. *ReLU* activation function was used for both the input and hidden layers and placed a dropout layer in between with a dropout probability of 0.005 for avoiding the overfitting problem. The size of the output layer was tuned to the unique number of classes of music genre classifiers.

Training the model over 50 epochs with an early stopping based on validation loss to avoid overfitting, training was done in batch sizes of 64 later tuned down to 32 for finer gradient updates. The *Adam* optimizer with an initial learning rate of 0.001 and applied L2 regularisation was implemented. The learning rate was controlled by the learning rate scheduler *ReduceLROnPlateau* in relation to validation performance.

3.2.2 Support Vector Machine (SVM)

For the SVM model, a general perspective was taken with respect to fine-tuning the performance. First the model was carefully tuned using an SVM RBF kernel, such that it might be able to pick up on the intricate patterns within the dataset. Using the *GridSearchCV*, the best hyperparameters were extracted for the fine-tuning of the model. This process involved an extensive search including ranges of values of the regularisation parameter C that would explore a fine balance between model complexity and generalisation. In the experimental stage, different types of kernel were tested, including RGF, lineal, and poly, to determine the best one to capture most subtleties within the data. Finally, the gamma parameter, crucial for the RBF and poly kernels, was tested both on auto and scale to optimise the decision boundary influenced by individual data points. The polynomial degrees of the polynomial kernel functions were varied to test the performance of the model at different complexities. This parameter tuning, through systematic exploration using 3-fold cross-validation, was an optimization task by intention but a strategic effort directed at strengthening the accuracy of the SVM model and its soundness in the correct classification of music genres.

4. Evaluation and Comparison of the Models

4.1 Multilayer Perceptron: Optimization and Results

A meticulous approach was adopted in optimising the MLP model, starting with the implementation of a basic model to establish a benchmark for accuracy. This strategy done for the improvement of the model involved a thorough review and iterative adjustments to several hyperparameters of the models. This included altering the network structure by varying the number of hidden layers and neurons to find the optimal balance between capacity and complexity; adjusting the learning rate through grid search to identify the ideal speed of weight adjustment that balanced efficiency and accuracy while avoiding local low point; and utilising advanced weight initialisation techniques that work well to the model's activation functions, to optimise the starting point for learning. Furthermore, regularisation and dropout techniques were incorporated to mitigate overfitting, allowing the model to maintain its performance on unseen data.

Figure 3 shows the loss values over epochs for the baseline model and the optimised one. The baseline graph displays the decrease for the training loss in a monotonically manner to values close to zero, and on the other hand, the test loss is noisier, especially at higher epochs. This kind of variability in the test loss could point toward the model possibly overfitting the training data and, therefore, might not generally reflect well on unseen data, even though it

showed a slightly higher accuracy of 58%. In contrast, the optimised model shows the training and test losses both fall much more linearly so that they would converge to better generalisation with new data. This is an indicative sign that the model captures the underlying patterns in the music features without overfitting and with an increase of accuracy with and end result of 59% around the 23rd epoch.

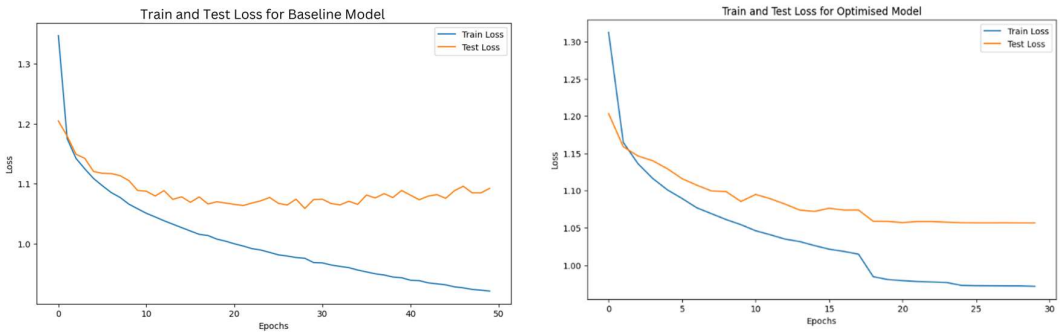

Figure 3. Loss plots for test and training data for the baseline MLP model and optimised MLP model.

4.2 Support Vector Machine: Optimization and Results

Similarly, for the SVM model a baseline model with the default configurations was used to determine the base accuracy that will be used to compare with the optimised model's accuracy and performance. The baseline model was compiled without any hyperparameters tuning, and this model achieved an initial accuracy of 57.63%.

The SVM optimization was approached with two separate strategies: grid search and Optuna. The grid search focused on systematically varying the regularisation parameter C, the kennel type (exploring options such as RBF, linear, and polynomial), and the gamma parameter for the RBF kernel. This was exhaustive, and required a large search space for the hyperparameters, having some computational cost, but offered a very systematic way of finding some best-performing hyperparameter combinations that improved model accuracy.

The optimization with Optuna provided a more effective, resultative approach, allowing more guided and focused search of hyperparameters. Optuna optimised the same hyperparameters as grid search and focused search of hyperparameters, tuning its search based on the results of past iterations to converge more rapidly towards the best settings. This technique not only cut down the search time but also led to a combination that gave an increased accuracy percentage, being 57.69%.

Classification Report for the Baseline Model

	Precision	Recall	f1-score	Support
Alternative	43.95%	37.12%	40.25%	1498
Anime	74.07%	71.55%	72.79%	1529
Blues	61.50%	54.50%	57.79%	1545
Classical	82.52%	84.92%	83.71%	1479
Country	53.43%	56.31%	54.83%	1481
Electronic	65.62%	59.93%	62.65%	1535
Hip-Hop	45.70%	53.42%	49.26%	1492
Jazz	54.45%	51.52%	52.95%	1483
Rap	45.67%	36.73%	40.71%	1492
Rock	50.27%	70.53%	58.70%	1466
Accuracy	57.63%	57.63%	57.63%	57.63%
Macro avg	57.72%	57.65%	57.36%	15000
Weighted avg	57.79%	57.63%	57.39%	15000

Classification Report for the Optuna Model

	Precision	Recall	f1-score	Support
Alternative	44.11%	37.52%	40.55%	1498
Anime	74.30%	71.48%	72.87%	1529
Blues	61.50%	54.50%	57.79%	1545
Classical	82.62%	85.19%	83.89%	1479
Country	53.42%	56.38%	54.86%	1481
Electronic	65.81%	60.07%	62.81%	1535
Hip-Hop	45.76%	53.22%	49.21%	1492
Jazz	54.58%	51.45%	52.97%	1483
Rap	45.56%	36.80%	40.71%	1492
Rock	50.22%	70.53%	58.67%	1466
Accuracy	57.69%	57.69%	57.69%	57.69%
Macro avg	57.79%	57.71%	57.43%	15000
Weighted avg	57.86%	57.69%	57.46%	15000

Figure 4. Table comparing the results between the SVM baseline model and the Optuna model for SVM.

5. Conclusion

Comparing the MLP and SM models within the music genre classification dataset, it brings to light distinct strengths and limitations within both methods. The architecture of the MLP neural network model is highly flexible and adapts well to the dataset to capture complex, non-linear relations between the values in the dataset. Even though from the beginning they evidenced overfitting, the applied optimizations have managed to reduce this problem to a 57% accuracy. On the other hand, the SVM model, with detail in the search strategy of hyperparameters, realised a bit higher accuracy of 57.69% and clearly identified itself through the effectiveness to define exact classification margins, even with complicated feature spaces.

The two models are quite close in performance to each other. Both models have an accuracy rate close to 57% for music genre classification, but the SVM model showed a higher potential to handle very high dimensionality and diversity of music features, while the optimised model required much less time with reduced computational effort. In this context, therefore, while MLP promises a solid theoretical basis and applicability in music data modelling, SVM, by its very nature of accuracy and efficiency, will be the best technique to perform music genre classification in this study.

References

[1] L. Peralta, “Impact of Music on Society - Sociological Effects,” *Save The Music Foundation*, Nov. 03, 2021. https://www.savethemusic.org/blog/how-does-music-affect-society/

[2] Popescu, Popescu, Balas, and Mastorakis, “Multilayer perceptron and neural networks,” *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 7, Art. no. 1109–2734, Jul. 2009, [Online]. Available: https://shorturl.at/jqIN2

[3] A. Kirchner and C. S. Signorino, “Using support vector machines for survey research,” *Survey Practice*, vol. 11, no. 1, pp. 1–14, Jan. 2018, doi: 10.29115/sp-2018-0001.

[4] “Prediction of music genre,” *Kaggle*, Nov. 02, 2021. https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre

Supplementary Material

1. Glossary

i. *Neural Computing*: “Multidisciplinary research in theory, modelling, computation, and statistics in neuroscience and in the design and construction of neutrally inspired information processing systems.” [1]

ii. *Weight*: Number that correspond to the connections between neurons, or nodes, across different layers of the network. Every connection has a weight assigned to it, which indicates the strength and direction of the influence between neurons. [2]

iii. *Forward Function*: “Represents the pathway through which data flows from the input layer, hidden layers and lastly the output layer. This process makes the predictions or classifications based on the input it receives.” [3]

iv. *Backpropagation*: Seeks to reduce the cost function by modifying the biases and weights inside the network. [4]

v. Pipeline: Connected set of data processing and modelling operations that is designed to automate, standardize and streamline the process of building, training, evaluation and deploying machine learning models. [5]

vi. *Grid Search*: “Hyperparameter tuning technique that performs an exhaustive search over a specified hyperparameter space to find the combination of hyperparameters that yields the best model performance.” [6]

vii. *Optuna*: “An open source hyperparameter optimization framework to automate hyperparameter search.” [7]

2. Intermediate results: Deep analysis of the optimizing results for both models

2.1. MLP

Training the MLP over 50 epochs showed a consistent reduction in loss and increase in accuracy, evident in both the training and test sets. Initially, the model began with a training loss of 1.35 and an accuracy of 50%, improving steadily to a training loss of 0.92 and an accuracy of 64% by the end of the training period. Meanwhile, the test set loss decreased from 1.20 to 1.09, with test set accuracy slightly rising from 0.53 to 0.57.

This progression indicates effective learning, though there's still room for enhancement. The convergence of loss and accuracy stabilization suggests that the model might be nearing its maximum learning capacity given its current architecture and hyperparameters. The test accuracy, which remained around 58% for most of the training, signifies a balance between learning and generalization. The final accuracy of 57% hints at a minor decline in generalization capability.

2.2. SVM

During the SVM model hyperparameter optimization process, an expository search was conducted in a space of 54 candidate configurations, including 162 settings: the combination of parameters C, degree, gamma, and kernel with three folds for each candidate. The execution times varied in the order of some magnitudes according to the configuration, pointing out the influence of kernel choice and the values assumed by C, gamma, and degree on the computer model complexity. Configurations using RBF and poly incurred higher runtime

than those using the linear kernel. This is indicative of an increased computational burden. Furthermore, the change of gamma between scale and auto, and the adjustment of the C parameter from small values to larger values do reflect an effort to balance the trade-off between the capacity for generalization and fitting to the training data.

The best configuration was found to be {'C': 1, 'degree': 2, 'gamma': 'scale', 'kernel': 'rbf'}, with a score of 56.9%, pointing to moderate This finding underscores the importance of the selection of hyperparameters in the performance of the SVM model, particularly in complex classification tasks in which high accuracy is key. The RBF kernel, on the other hand, brought forth its capacity of nonlinear feature space handling, therefore providing an elastic model with respect to data structure. However, the obtained score also might indicate the limitations of the model's ability to effectively capture data variability or may be further explored through strategies related to data pre-processing or even with feature engineering improvements.

References for the Glossary

[1] “MIT Press Direct,” *Mit.edu*, 2024. https://direct.mit.edu/neco

[2] DeepAI, “Weight (artificial neural network),” *DeepAI*, May 17, 2019. https://deepai.org/machine-learning-glossary-and-terms/weight-artificial-neural-network

[3] “Feedforward Neural Network,” Deepgram, Feb. 16, 2024. https://deepgram.com/ai-glossary/feedforward-neural-network

[4] S. Kostadinov, “Understanding Backpropagation Algorithm,” Medium, Aug. 08, 2019. https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd

[5] “What is a Machine Learning Pipeline? | IBM,” www.ibm.com. https://www.ibm.com/topics/machine-learning-pipeline

[6] C. Bala Priya, “Hyperparameter Tuning: GridSearchCV and RandomizedSearchCV, Explained,” KDnuggets, Nov. 03, 2023. https://www.kdnuggets.com/hyperparameter-tuning-gridsearchcv-and-randomizedsearchcv-explained

[8] “Optuna - A hyperparameter optimization framework,” Optuna. https://optuna.org