Reduced Convolutional Recurrent Neural Network Using MFCC for Music Genre Classification on the GTZAN Dataset

Ela Setiorini, Moeljono Widjaja, Arya Wicaksana*
Universitas Multimedia Nusantara, Tangerang 15810 Indonesia
E-mail: ela.setiorini@student.umn.ac.id, moeljono.widjaja@umn.ac.id, arya.wicaksana@umn.ac.id
*Corresponding author

Keywords: classification, CRNN, GTZAN, MFCC, MIR, music genre

Received: August 9, 2024

This study presents a reduced Convolutional Recurrent Neural Network (CRNN) model for music genre classification, leveraging the GTZAN dataset and Mel-Frequency Cepstral Coefficient (MFCC) feature extraction. Unlike more complex architectures, this model simplifies the CRNN structure to three convolutional layers and two BiLSTM layers, maintaining competitive performance while reducing computational complexity. Key experimental parameters included learning rate tuning (0.1, 0.01, 0.001, and 0.0001) and dropout usage (30% before the BiLSTM layers) to mitigate overfitting. The best configuration, utilizing a learning rate of 0.001 and dropout, achieved an accuracy of 88.64%, outperforming more complex CRNN models by approximately 15%. These results underscore the potential of streamlined architectures in music information retrieval tasks, particularly for applications where computational resources are constrained. Future work will address overfitting issues and refine the dataset for enhanced model performance.

Povzetek: Študija predstavlja poenostavljen model konvolucijsko-rekurentne nevronske mreže (CRNN) za klasifikacijo glasbenih zvrsti z uporabo GTZAN podatkovne zbirke in MFCC značilk, ki kljub zmanjšani kompleksnosti dosega visoko natančnost (88,64 %) ter presega zmogljivejše modele za približno 15 %.

1 Introduction

Music genre classification is crucial for information retrieval (Music Information Retrieval) and analysis, and as digital music libraries grow, automated methods are needed to categorize and organize music. Music Information Retrieval (MIR) is a research field that focuses on analyzing and extracting music transcription, beat detection, on-set detection, and genre classification [1]. Traditional methods often rely on handcrafted features and machine learning algorithms, which need help to capture complex temporal and spectral patterns. Convolutional Recurrent Neural Networks (CRNNs), a combination of CNNs for feature extraction and RNNs for temporal dependencies, have shown promising results in extracting hierarchical features from raw audio data [2].

This work investigates further the CRNN application for MIR, specifically to classify music genres with the GTZAN dataset. The CRNN extracts local features and aggregates temporal patterns [3]. MFCC is extracted from the dataset to become an input for the model [4]. The combination of CRNN in this work is CNN with BiLSTM (Bidirectional Long Short-Term Memory). This work proposes a simpler version of the algorithms compared to Ashraf et al. [5], with only three layers of CNN and two layers of RNN (BiLSTM-BiLSTM). The accuracy of Ashraf et al. is 73.69% with five layers of CNN and three layers of RNN. The main contribution of this work is a less complex CRNN model architecture with higher accuracy.

The rest of this paper is structured as follows: The Introduction section sets the background and motivation for this paper. Related Works section layouts other works in MIR that utilize machine learning and deep learning methods. The Methods section explains the data collection, requirement specification, design and implementation, and testing and evaluation. The Results and Analysis section presents empirical evidence based on four scenarios and discusses the findings. Finally, the Conclusion section summarizes the findings and outlines future works.

2 Related works

Ghosh et al. [2] compared machine learning models (SVC or Support Vector Classifier, logistic regression, and ensemble learning using AdaBoost) and deep learning models (ANN or Artificial Neural Network, CNN, CRNN with CNN-LSTM combination, and PCRNN or Parallel CRNN). Inputs used in this work are feature matrix (for machine learning models) and Mel-spectrogram (for deep learning models), extracted from the FMA dataset. The CRNN has the highest accuracy at 90%, with only 480 out of 8000 data used from the dataset.

Ashraf et al. [5] use a couple of CRNN combinations, i.e., CNN-LSTM, CNN-BiLSTM, CNN-GRU, and CNN-BiGRU. Inputs used in this work are the Mel-spectrogram and Mel-Frequency Cepstral Coefficient (MFCC) extracted from the GTZAN dataset. This work's highest

accuracy is obtained by CNN-BiGRU using Mel-spectrogram (89.3%) and CNN-LSTM using MFCC (76.4%). Mendes et al. [6] compared two CRNN combinations (CNN-LSTM and CNN-BiLSTM), and the results were used to get music recommendations. The input that is used in this work is the Mel-spectrogram that is extracted from the FMA dataset. CNN-BiLSTM achieved the highest accuracy, with an accuracy of 72%.

Luo [7] compared two deep learning algorithms, Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), using GTZAN and FMA (Free Music Archive) datasets. The accuracies obtained for the GTZAN dataset are 56% (CNN) and 42% (LSTM). Meanwhile, accuracies for the FMA dataset are 50.5% (CNN) and 33.5% (LSTM). Kumar et al. [8] used CNN and GTZAN datasets to classify music genres. The accuracy obtained in this work is 83%. Ghosh et al. [2] compared several machine learning and deep learning algorithms using the FMA dataset. The highest accuracy obtained in this work was achieved using a convolutional-recurrent neural network (CRNN) with an accuracy of 90%. Table 1 provides a summary and comparison of these related works.

Table 1: Test results from the model

Work	Model	Dataset	Feature Extraction	Accuracy
Luo [7] – "Automatic Music Genre Classification based on CNN and LSTM"	- CNN - LSTM	- GTZAN - FMA	- Mel-spectrogram (for CNN) - MFCC (for LSTM)	- Highest accuracy obtained by CNN using both GTZAN dataset (56%) - FMA dataset (50.5%)
Kumar et al. [8] – "Automated Music Genre Classification through Deep Learning Techniques"	CNN	GTZAN	MFCC	Accuracy obtained by the model is 83%
Ghosh et al. [2] – "A Study on Music Genre Classification using Machine Learning"	- Machine learning models (SVC, logistic regression, and ensemble learning using AdaBoost) - Deep learning models (ANN, CNN, CRNN with CNN-LSTM combination, PCRNN)	FMA, using only 480 out of 8000 data	- Feature matrix (for machine learning models) - Mel-spectrogram (for deep learning models)	Highest accuracy obtained by CRNN (90%)
Ashraf et al. [5] – "A Hybrid CNN and RNN Variant Model for Music Classification"	- CNN-LSTM - CNN-BiLSTM - CNN-GRU - CNN-BiGRU	GTZAN	- Mel-spectrogram - MFCC	- Highest accuracy obtained by CNN-BiGRU using Mel-spectrogram (89.3%) - CNN-LSTM using MFCC (76.4%)
Mendes et al. [6] – "Deep Learning Techniques for Music Genre Classication and Building a Music Recommendation System"	- CNN-LSTM - CNN-BiLSTM	FMA	Mel-spectrogram	CNN-BiLSTM obtained the highest accuracy (90%)

3 Methods

The dataset that is used in this work is the GTZAN dataset. GTZAN dataset was first created by George Tzanetakis and Perry Cook in 2002 [9]. This dataset consists of 10 genres of music, and each genre has a total of 100 WAV audio. Each audio file lasts 30 seconds with a 22,050 Hz sample rate. The genres inside this dataset are blues, classic, country, hip-hop, jazz, metal, pop, reggae, and rock. There is a single corrupted audio file within the jazz category. `The corrupted audio within the jazz category is removed before it got into data processing. Hence, the dataset contains only 999 audio files. The dataset was initially available on a MARSYAS website created by Tzanetakis and Cook [10]. Currently, the dataset is available to download at https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification (Kaggle).

MFCC is extracted from each audio file inside the dataset. Sound extraction carried out by MFCC is based

on estimated frequencies that humans can hear. The signal used in MFCC is the Mel scale, which uses a linear filter at frequencies below 1000 Hz and a logarithmic distance above 1000 Hz [11]. The output of this process is a spectrum wave graph or spectrogram that uses this frequency scale. This spectrogram contains feature coefficients, and these coefficient values represent the audio signal [12]. MFCC can capture important voice characteristics in recognition and critical information in voice, produce minimal data without losing much information, and replicate human hearing sound [13].

MFCC extraction is handled using Librosa, a Python package that provides audio and music signal processing [14]. Several parameters are needed to extract MFCC, such as the number of coefficients needed, window length, and hop length. This work extracts 13 cepstral coefficients from every audio. Window length and hop length used are 2,048 and 512, respectively. Each audio file is split into ten segments to augment the training data. The results of this process are saved on a JSON file containing an array with the 13 MFCC coefficient inside.

The proposed CRNN model consists of three layers: a convolutional layer, a max pooling layer, and batch normalization on the CNN layer, with two BiLSTM layers for the RNN. After going into the CNN and RNN layers, the inputs are flattened, and the fully connected layer connects all the neurons with the output layer. This work investigates two models: the models that used dropout before the BiLSTM layer and those that did not use dropout. The CNN layer consists of a convolutional layer, a max pooling layer, and batch normalization. The convolutional layer uses 32 filters with 3x3 kernel size on the first and second layers and 2x2 kernel size on the third layer, including ReLU for the activation function. ReLU is preferred in this work due to its ability to remove negative values.

The Max pooling layer has a 3x3 pool size for the first and second layers, 2x2 for the third layer, and 2x2 stride. Pooling is done to progressively lower the model's computational complexity, parameter count, and control overfitting [15]. Pooling reduces the size of the matrix in the feature map. Max pooling is one of the most popular forms of pooling. Max pooling extracts the highest value inside patches from the feature map and discards the rest of the values [16].

Batch normalization is employed to mitigate sudden changes in each layer [6]. The input is reshaped before being passed to the RNN layer, as the CNN and RNN layers require different input shapes. In this work, a dropout layer is added before the RNN layer as part of the investigation. Dropout is widely recognized for its ability to prevent overfitting by randomly deactivating neurons during training [17]. A dropout rate of 0.3 (30%) is used before the RNN layer. This value is selected based on its superior validation accuracy compared to dropout rates of 0.2 and 0.4.

In this work, two layers of BiLSTM are used, with return sequences set to true for the first layer. Return sequences are used to send forward all of the LSTM hidden layer sequences to the next layer [18]. The input went to the dense or fully connected layer. The input is flattened before going to the dense layer. The number of filters used for the dense layer is 64 units with the ReLU activation function, which is then followed by another dropout of 30%. Figure 1 shows the overview of the model proposed in this work.

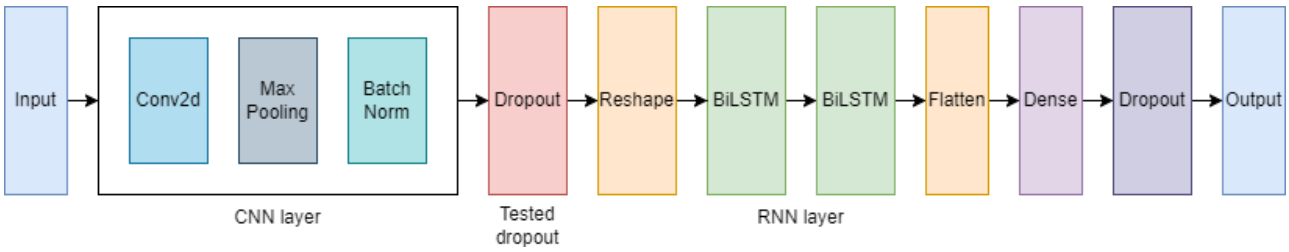


Figure 1: Architecture overview of the proposed CRNN model.

The dataset is divided into training, validation, and test sets with a 60-20-20 ratio using a simple train-validation-test split. Initially, the dataset is split into training and test sets with an 80-20 ratio. The 80% training set is then further divided into training and validation sets with a 75-25 ratio. The validation set is used to evaluate the model during training, while the test set is reserved exclusively for testing the trained model and is not used during the training process.

Several parameters, such as the optimizer, loss function, and evaluation metrics, are configured for the model. The optimizer employed in this work is Adam, tested with multiple learning rates: 0.1, 0.01, 0.001, and 0.0001. These learning rates were selected to observe the accuracy trend, specifically whether smaller learning rates lead to higher accuracy. The loss function is sparse categorical cross-entropy, and the primary performance metric is accuracy.

The model is trained for 100 epochs with a batch size of 32. After training, the model is evaluated using the test set. The evaluation metrics include accuracy, loss, precision, recall, and the F1 score.

4 Results and analysis

This work tests several parameters, namely dropout before the RNN layer and several learning rates values. Model testing uses the test set, which has not been used in any process. Table 2 shows the model's testing result using the test set.

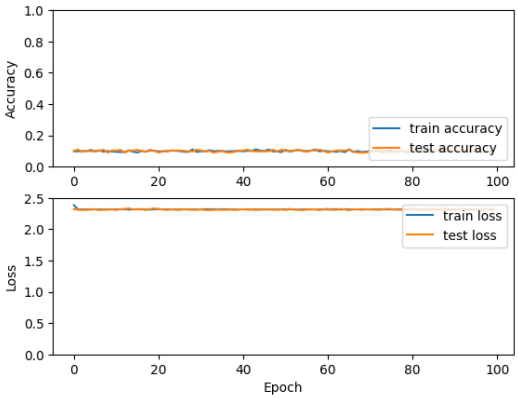
Table 2: Test results from the model

Dropout	Learning rate	Accuracy	Loss	Precision	Recall	F1 score
-	0.1	10.06%	230.86%	1.01%	10.06%	1.84%
-	0.01	74.07%	91.52%	74.82%	74.07%	73.81%
-	0.001	86.74%	64.69%	86.84%	86.74%	86.67%
-	0.0001	84.68%	60.25%	84.7%	84.68%	84.56%
✓	0.1	10.91%	230.88%	1.19%	10.91%	2.15%
✓	0.01	74.72%	75.5%	74.94%	74.72%	74.43%
✓	**0.001**	**88.64%**	**51.95%**	**88.85%**	**88.64%**	**88.62%**
✓	0.0001	86.24%	46.91%	86.28%	86.24%	86.19%

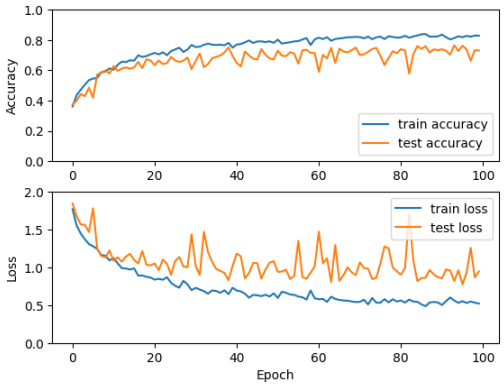
Both models, with or without dropout that used a 0.1 learning rate had very low accuracy because the learning rate was too big. A significant learning rate would speed up the training process, but the model could need more time to analyze the data thoroughly. Based on the accuracy, models with a learning rate of 0.1 could not study the data. Models with a 0.01 learning rate had similar accuracy, around 74%. Models using a 0.001 learning rate had the highest result out of all learning rates, but when compared to the use of dropouts, the model using dropouts had higher accuracy.

The model that used dropout and a 0.001 learning rate got the highest accuracy out of all other tested models, with an accuracy of 88.64%. Models with a 0.0001 learning rate achieved high results, but not higher than models with a 0.001 learning rate. Loss results from models that used dropout are significantly lower than those that did not. Meanwhile, each model's precision, recall, and F1 scores have similar results in terms of accuracy.

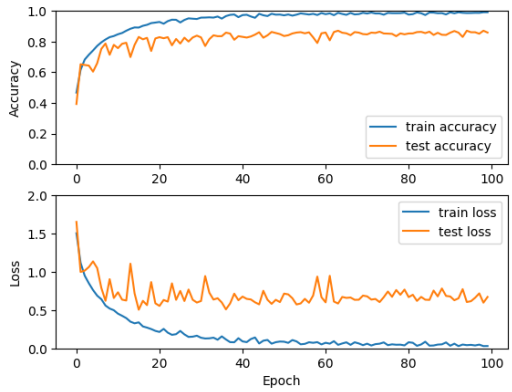
The following figures are accuracy and loss graphs from the training process. The models suffer from overfitting, but models that used dropout suffered less than those that did not. Figure 2 shows training results from models without dropouts, and Figure 3 shows training results from models with dropouts. The blue line represents results from the train set, and the orange line represents results from the validation set.



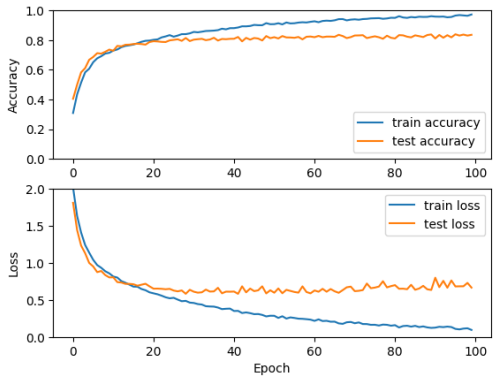
(a)



(b)

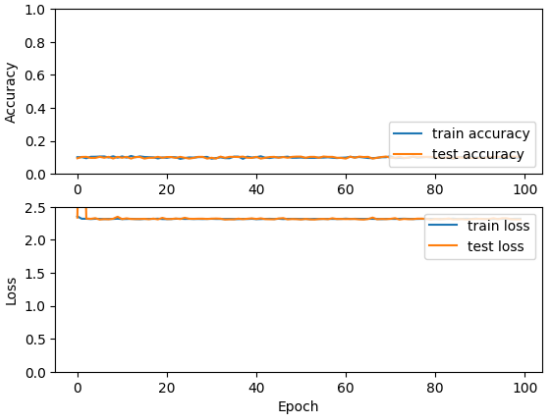


(c)

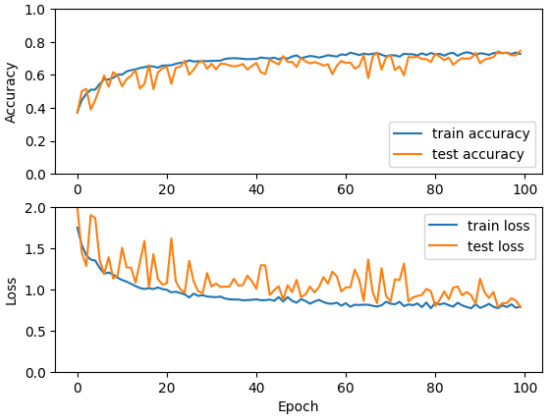


(d)

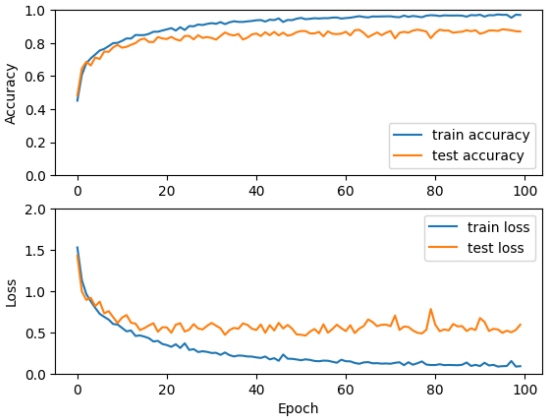
Figure 2: Accuracy and loss graph for models without dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001



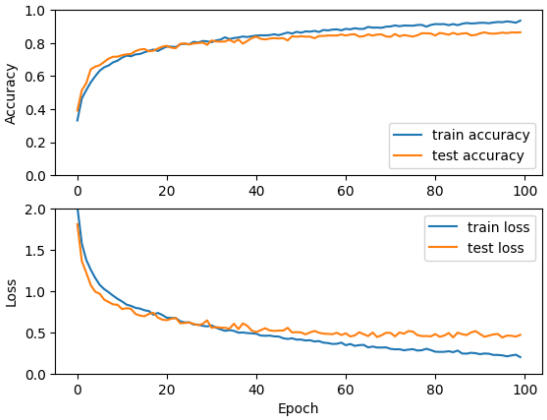
(a)



(b)



(c)



(d)

Figure 3: Accuracy and loss graph for models with dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001

Based on Figures 2 and 3, graphs from learning rate 0.1 show underfitting due to the high loss and low accuracy achieved on all epochs. The results indicate that the model is unable to learn from the data. A learning rate of 0.01 shows overfitting only for the model without dropout. The other model had a good fit because the difference between the train and validation set is minimal. Learning rates of 0.001 and 0.0001 also show overfitting, but the model with dropout is better than without dropout. Thus, the use of dropout effectively reduces overfitting on models. Weight regularization such as L2 are not implemented in this model. Only dropout is used in this

model to maintain the simplicity of the model and also due to device limitation.

Further evaluation on Figures 4 and 5 depict the confusion matrix of the results. Figure 4 shows the confusion matrix from models without dropouts, and Figure 5 shows the confusion matrix from models with dropouts. The confusion matrix for models with a learning rate of 0.1 showed that the model could not classify any data. Meanwhile, models with a learning rate of 0.01 made many mistakes when classifying data. The accuracy achieved using this learning rate is relatively low, around 70%. Learning rates 0.01 and 0.001 show overfitting but are still able to classify the data sufficiently.

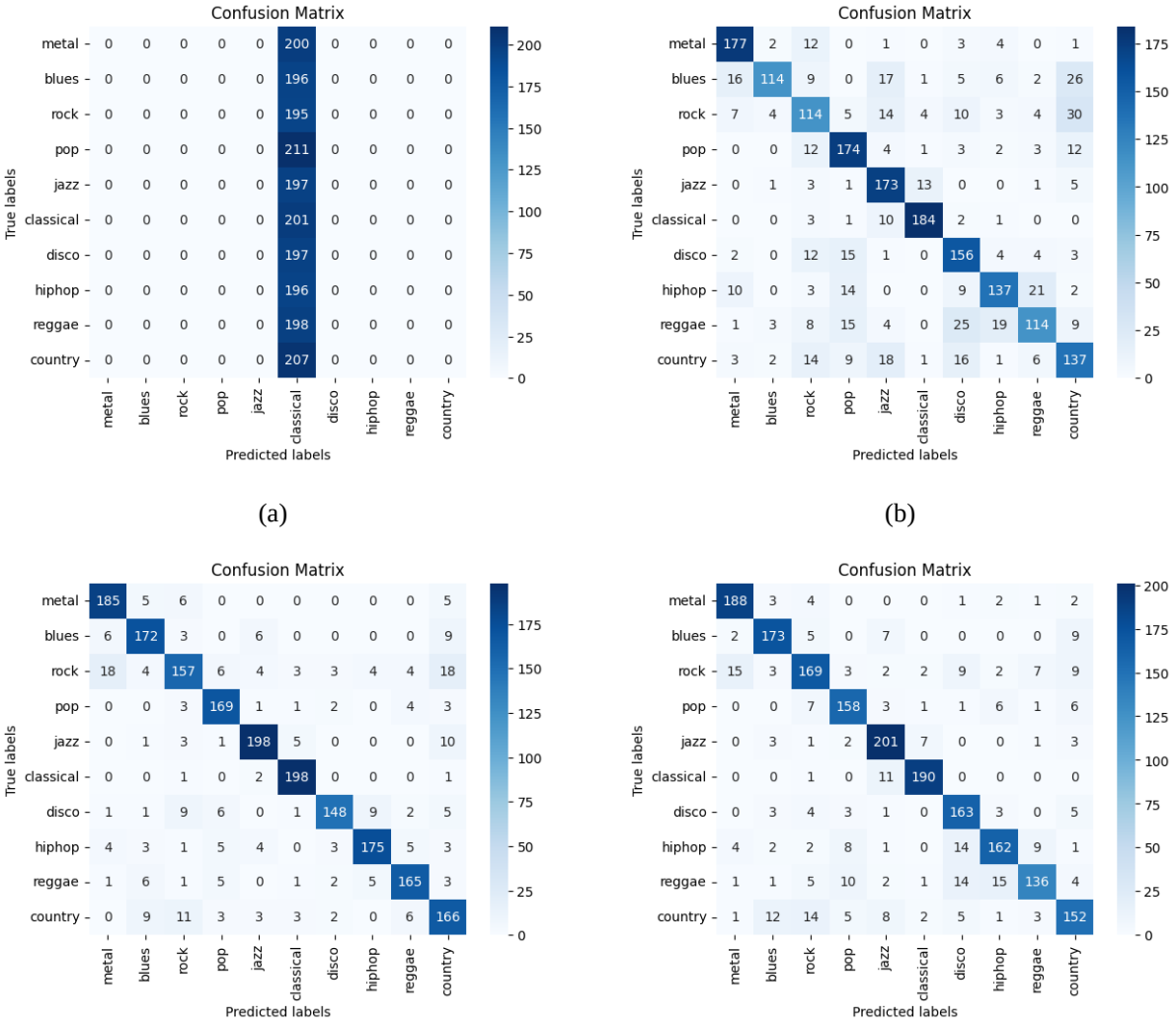


Figure 4: Confusion matrix of models without dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001

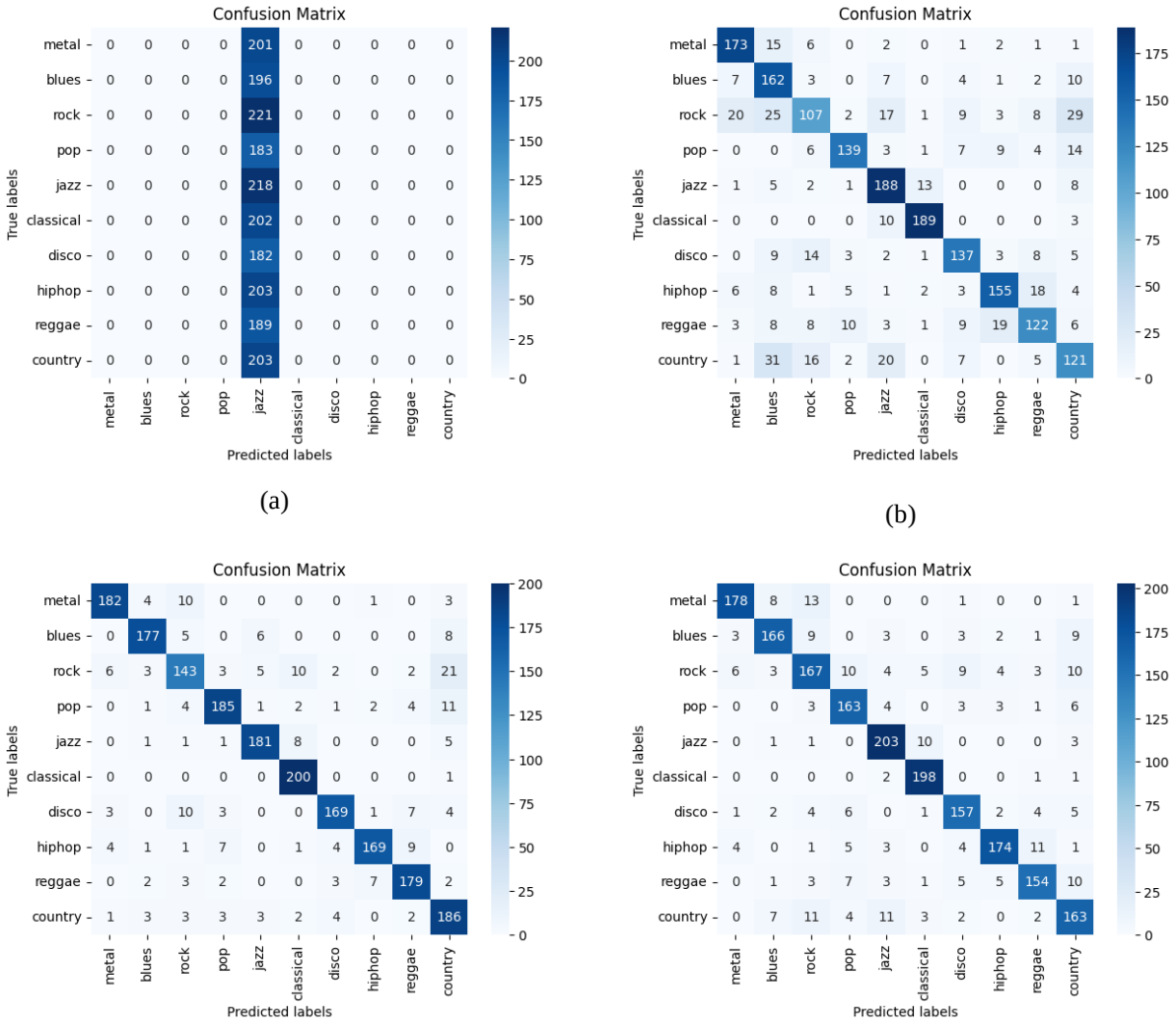


Figure 5: Confusion matrix of models with dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001

The ROC curves in Figures 6 and 7 provide detailed insights into the classification performance of the models under different configurations. Key metrics such as the true positive rate (TPR) and false positive rate (FPR) are visually represented, where a steeper curve rising toward the top-left corner indicates higher sensitivity (recall) and better classification of positive instances. Additionally, a lower FPR, reflected in curves closer to the y-axis, suggests improved discrimination between classes.

The area under the curve (AUC) serves as a summary metric, quantifying the model's overall performance. Higher AUC values, closer to 1.0, denote stronger classification capabilities. By comparing models with and without dropout across varying learning rates, it becomes evident that the inclusion of dropout generally enhances the stability and sharpness of the curves, indicating better generalization and resistance to overfitting. Notably, models with lower learning rates (e.g., 0.001 and 0.0001) demonstrate more consistent and pronounced ROC curves, suggesting that these configurations strike a balance between convergence and classification accuracy.

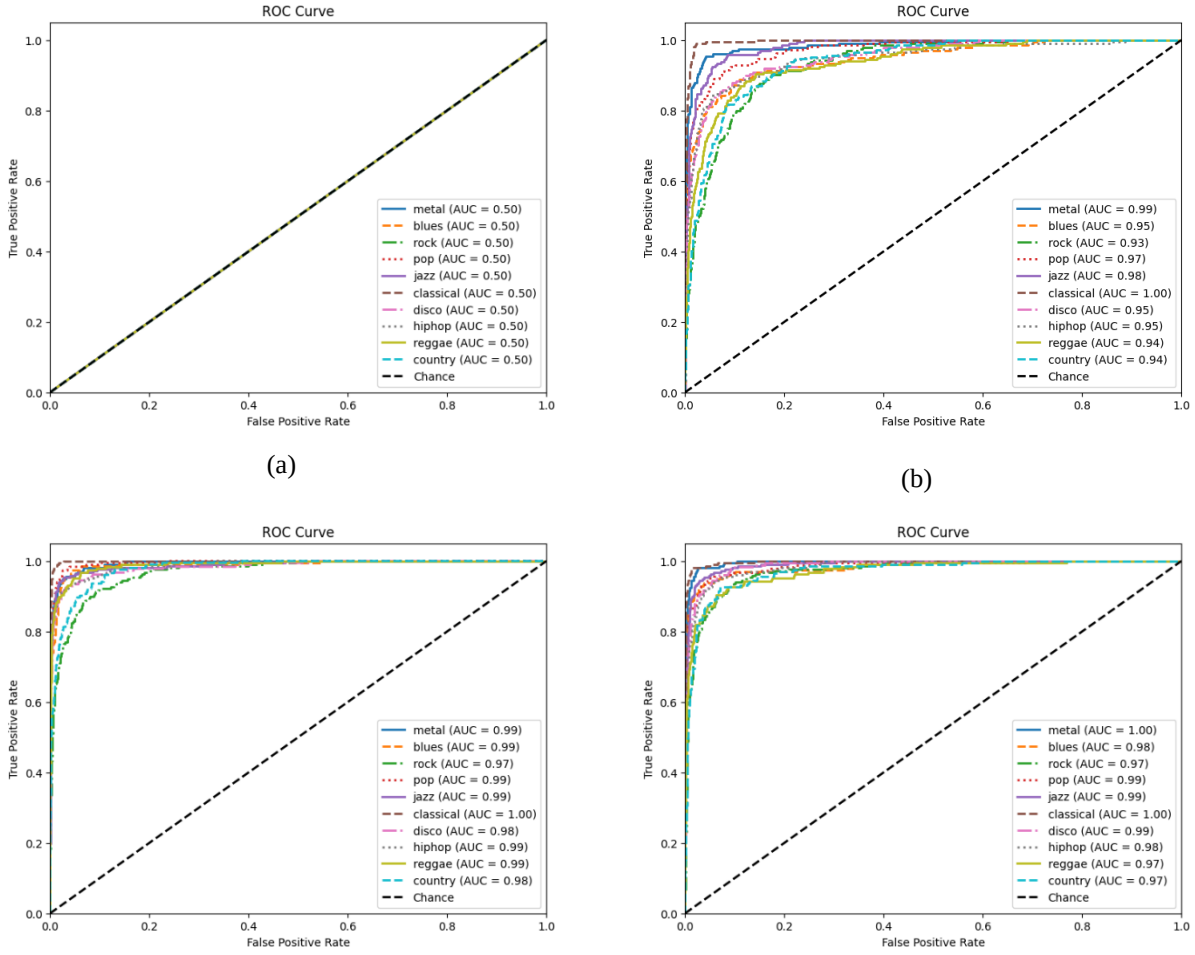
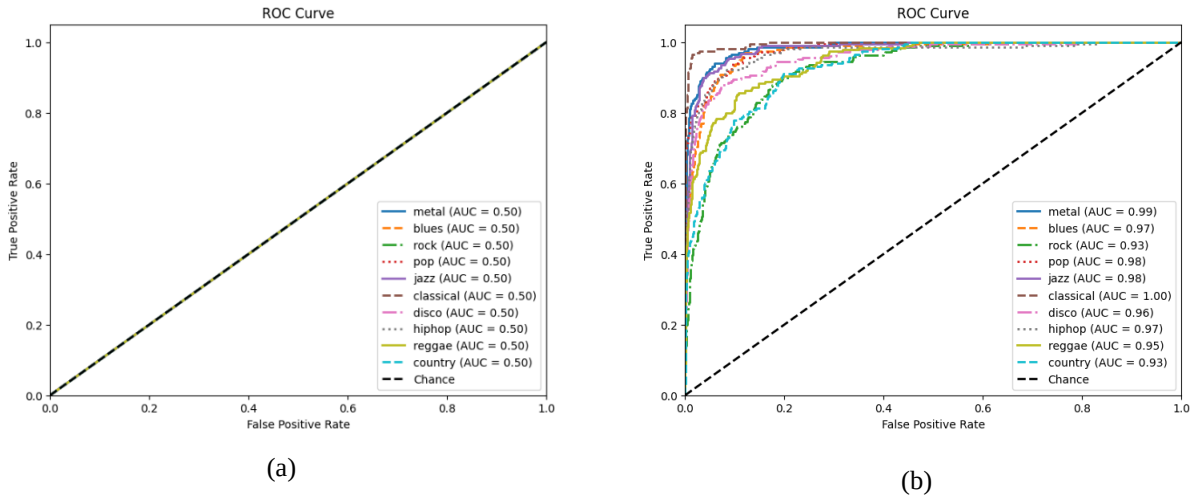
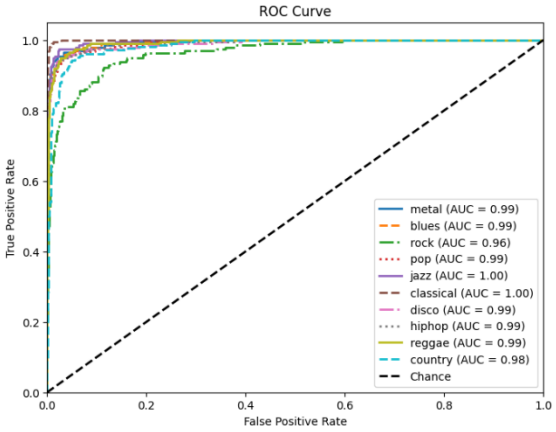
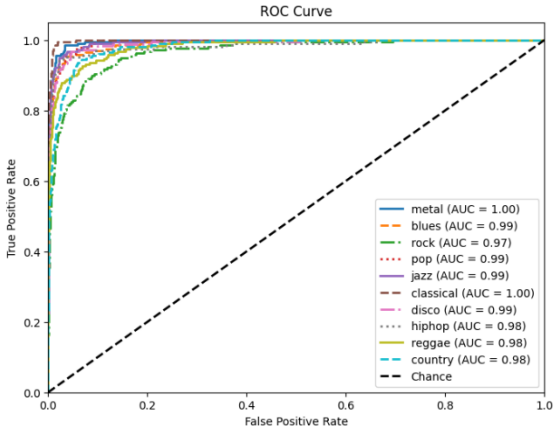


Figure 6: ROC curves of models without dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001





(c)



(d)

Figure 7: ROC curves of models with dropout with (a) Learning rate 0.1, (b) Learning rate 0.01, (c) Learning rate 0.001, and (d) Learning rate 0.0001

A confusion matrix of a test set with 1,998 data points shows that the model with dropout and learning rates 0.01, 0.001, and 0.0001 can classify 1,493, 1,771, and 1,723 data, respectively. In contrast, the model without dropout can classify 1,480, 1,733, and 1,692 data out of 1,998. The model with a learning rate of 0.1 will not be discussed further because the model cannot classify the data. The model using dropout and learning rate 0.001 has the highest number of correct classification results compared to the other models, indicating that it is the best model.

Further analysis of the results indicates that the classical genre is the most accurately classified genre. The rock genre is the most misclassified due to the proximity between rock, country, and disco. The dataset also poses discrepancies with other references regarding the genre of certain audio files. In addition, the lack of artist variation in the dataset also affects the model classification ability due to the lack of variation of music genres, as an artist tends to produce music within one music genre.

Audio from the rock genre was frequently misclassified as country, disco, or blues. Additionally, the model often misclassified non-rock audio as rock. Upon closer examination of the dataset, this issue appears to stem from the presence of numerous repetitive patterns and mislabeled samples under the rock category.

The findings of this study have significant implications for the field of music information retrieval (MIR), particularly in advancing genre classification systems. The results demonstrate that a simpler CRNN architecture, when paired with effective regularization techniques like dropout and optimized learning rates, can achieve competitive accuracy, outperforming more complex models while maintaining computational efficiency. The highest classification accuracy of 88.64% highlights the potential of lightweight models in resource-constrained environments.

Moreover, the analysis underscores critical challenges in MIR, such as dataset quality and diversity. Issues like mislabeled samples, genre overlaps, and limited artist variation directly impact classification performance, as evidenced by the frequent misclassification of rock as similar genres like country, disco, and blues. These challenges emphasize the importance of curated datasets and robust preprocessing techniques in developing reliable MIR systems. Addressing these limitations, along with strategies like artist-level cross-validation and stratified sampling, could not only enhance genre classification accuracy but also extend the applicability of MIR systems to more nuanced tasks, such as personalized music recommendations and musicological analysis.

5 Conclusion

This work proposes a simpler architecture of a CRNN algorithm to classify music. The aim is to see whether a greater accuracy is achieved compared to more complex models. The proposed model comprises three CNN layers and two RNN (specifically BiLSTM) layers. The feature extraction used in this model is MFCC, which splits the dataset into ten segments to increase the total data. A dropout level of 0.001 learning rate achieved the highest accuracy of 88.64%. The accuracy is higher than the previous work accuracy of 73.69% using the same CRNN model with more CNN and RNN layers.

The primary limitation of this work is that the model tends to overfit, necessitating further efforts to mitigate this issue. Future work could focus on cleansing the dataset by removing redundant and incorrectly labeled songs, which may enable the model to learn more effectively from the data. Additionally, incorporating early stopping techniques could help prevent overfitting. Other potential strategies include applying weight regularization, implementing artist-level cross-validation, and utilizing stratified sampling to improve the model's robustness.

Acknowledgements

The authors express gratitude to Universitas Multimedia Nusantara for the support and acknowledge the valuable input of the reviewers and associate editor.

References

[1] D. Stefani and L. Turchet, “On the Challenges of Embedded Real-time Music Information Retrieval,” in *Proceedings of the International Conference on Digital Audio Effects, DAFx*, 2022, pp. 177–184.

[2] P. Ghosh, S. Mahapatra, S. Jana, and R. Kr. Jha, “A Study on Music Genre Classification using Machine Learning,” *Int. J. Eng. Bus. Soc. Sci.*, vol. 1, no. 04, pp. 308–320, 2023, doi: 10.58451/ijebss.v1i04.55.

[3] J. Sang, S. Park, and J. Lee, “Convolutional recurrent neural networks for urban sound classification using raw waveforms,” *Eur. Signal Process. Conf.*, vol. 2018-Septe, pp. 2444–2448, 2018, doi: 10.23919/EUSIPCO.2018.8553247.

[4] V. Bella and S. A. Sanjaya, “Refining Baby Cry Classification using Data Augmentation (Time-Stretching and Pitch-Shifting), MFCC Feature Extraction, and LSTM Modeling,” in *2023 7th International Conference on New Media Studies (CONMEDIA)*, IEEE, 2023. doi: https://doi.org/10.1109/CONMEDIA60526.2023.10428158.

[5] M. Ashraf *et al.*, “A Hybrid CNN and RNN Variant Model for Music Classification,” *Appl. Sci.*, vol. 13, no. 3, 2023, doi: 10.3390/app13031476.

[6] J. Mendes, “Deep Learning Techniques for Music Genre Classification and Building a Music Recommendation System,” National College of Ireland, 2020.

[7] X. Luo, “Automatic Music Genre Classification based on CNN and LSTM,” *Highlights Sci. Eng. Technol.*, vol. 39, pp. 61–66, 2023, doi: 10.54097/hset.v39i.6494.

[8] M. K. Kumar, K. Sujanasri, B. Neha, G. Akshara, P. Chugh, and P. Haindavi, “Automated Music Genre Classification through Deep Learning Techniques,” *E3S Web Conf.*, vol. 430, 2023, doi: 10.1051/e3sconf/202343001033.

[9] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–302, 2002, doi: 10.1109/TSA.2002.800560.

[10] G. Tzanetakis and P. Cook, “MARSYAS: A framework for audio analysis”.

[11] S. Y. Yehezkiel and Y. Suyanto, “Music Genre Identification Using SVM and MFCC Feature Extraction,” *IJEIS (Indonesian J. Electron. Instrum. Syst.*, vol. 12, no. 2, p. 115, 2022, doi: 10.22146/ijeis.70898.

[12] I. D. G. Y. A. Wibawa and I. D. M. B. A. Darmawan, “Implementation of audio recognition using mel frequency cepstrum coefficient and dynamic time warping in wirama praharsini,” *J. Phys. Conf. Ser.*, vol. 1722, no. 1, 2021, doi: 10.1088/1742-6596/1722/1/012014.

[13] H. Heriyanto, T. Wahyuningrum, and G. F. Fitriana, “Classification of Javanese Script Hanacara Voice Using Mel Frequency Cepstral Coefficient MFCC and Selection of Dominant Weight Features,” *J. Infotel*, vol. 13, no. 2, pp. 84–93, 2021, doi: 10.20895/infotel.v13i2.657.

[14] B. McFee *et al.*, “librosa: Audio and Music Signal Analysis in Python,” *Proc. 14th Python Sci. Conf.*, no. August 2020, pp. 18–24, 2015, doi: 10.25080/majora-7b98e3ed-003.

[15] P. A. Aritonang, M. E. Johan, and I. Prasetiawan, “Aspect-Based Sentiment Analysis on Application Review using CNN (Case Study : Peduli Lindungi Application),” *Ultim. Infosys J. Ilmu Sist. Inf.*, vol. 13, no. 1, pp. 54–61, 2022.

[16] R. Yamashita, M. Nishio, R. Kinh, G. Do, and K. Togashi, “Convolutional neural networks : an overview and application in radiology,” pp. 611–629, 2018.

[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[18] A. Katrompas and V. Metsis, “Enhancing LSTM Models with Self-attention and Stateful Training,” in *Lecture Notes in Networks and Systems*, 2022, pp. 217–235. doi: 10.1007/978-3-030-82193-7_14.