Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer
Vol. 23, No. 2, March 2023, pp. 297∼308
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 200/M/KPT/2020
DOI: 10.30812/matrik.v23i2.3037

Learning Accuracy with Particle Swarm Optimization for Music Genre Classification Using Recurrent Neural Networks

Muhammad Rizki , Arief Hermawan , Donny Avianto
Universitas Teknology Yogyakarta, Yogyakarta, Indonesia

Article Info

Article history:

Received June 14, 2023
Revised 26 December, 2023
Accepted 31 January, 2024

Keywords:

Learning Accuracy
Music Classification
Particle Swarm Optimization
Recurrent Neural Networks

ABSTRACT

Deep learning has revolutionized many fields, but its success often depends on optimal selection hyper-parameters, this research aims to compare two sets of learning rates, namely the learning set rates from previous research and rates optimized for Particle Swarm Optimization. Particle Swarm Optimization is learned by mimicking the collective foraging behavior of a swarm of particles, and repeatedly adjusting to improve performance. The results show that the level of Particle Swarm Optimization is better previous level, achieving the highest accuracy of 0.955 compared to the previous best accuracy level of 0.933. In particular, specific levels generated by Particle Swarm Optimization, for example, 0.00163064, achieving competitive accuracy of 0.942-0.945 with shorter computing time compared to the previous rate. These findings underscore the importance of choosing the right learning rate for optimizing the accuracy of Recurrent Neural Networks and demonstrating the potential of Particle Swarm Optimization to exceed existing research benchmarks. Future work will explore comparative analysis different optimization algorithms to obtain the learning rate and assess their computational efficiency. These further investigations promise to improve the performance optimization of Recurrent Neural Networks goes beyond the limitations of previous research.

Corresponding Author:

Muhammad Rizki, +6282321451094
Faculty of Science and Technology, Department of Informatic,
Universitas Teknologi Yogyakarta, Yogyakarta, Indonesia
Email: 6220211002.rizki@student.uty.ac.id

How to Cite:
M. Rizki, A. Hermawan, and D. Avianto, ”Learning Accuracy with Particle Swarm Optimization for Music Genre Classification Using Recurrent Neural Networks”, *MATRIK: Jurnal Managemen,Teknik Informatika, dan Rekayasa Komputer*, Vol. 23, No. 2, pp. 297-308, Mar, 2024.

1. INTRODUCTION

Data mining is a well-developed method of extracting valuable insights from vast amounts of data. This process typically involves two main stages: data preparation and mining and the interpretation and analysis of the results obtained [1]. One data mining trend is utilizing the algorithms provided by Deep Learning. Deep learning has demonstrated impressive achievements in a range of machine learning assignments [2]; this is evidenced by the many cases that can be solved by Deep learning, such as estimation [3], prediction [4], classification [5], and clustering [6]. However, even though the scope of problems that can be solved by deep learning is vast, it is difficult to determine the right hyperparameters to apply to the machine consuming the Deep learning algorithm. One of the hyperparameters that needs to be tuned is the learning rate. The learning rate is a parameter that decides how much a model should adjust its parameters based on the error each time they are updated. Choosing the right learning rate is crucial because if it's too low, the model takes a long time to reach the best error point, and if it's too high, the model may deviate from the optimal error point [7, 8]. The learning rate is generally expected to fall between a range of values, typically from a bit over 10-6 to almost 1. However, for most networks, a value of 0.01 is the usual or standard value [9].

Several studies investigate the effect of learning rate on the model's accuracy; for example, the study conducted in 2021 aimed to determine the appropriate learning rate to be used in the Convolutional Neural Network (CNN) algorithm; the results of the study found that the CNN algorithm successfully classified images with an accuracy value of 99.4% using a learning rate of 0.0001 [10]. In addition, there was also a similar study carried out in 2023; the focus of that study was to investigate the conduct of different optimization algorithms that rely on gradients and are commonly employed in Deep Learning. That study also aims to compare the efficiency of these algorithms across various learning rates. In that study, the learning rate value is determined manually by selecting from a range of options, including 0.001, 0.003, 0.01, 0.03, and 0.1 [11]. Moreover, captivating insights about the learning rate have been unveiled through a comprehensive investigation in 2019. The study entailed a comparative analysis between the Back Propagation and Learning Vector Quantization (LVQ) approaches. The findings demonstrated that the backpropagation method achieved an impressive pattern recognition accuracy rate of 99.17% when utilizing a learning rate variation of 0.1 and an epoch of 100. Conversely, the learning vector quantization method exhibited a slightly lower accuracy rate of 96.67% with a learning rate variation of 1 and an epoch of 20 [12]. The findings mentioned above indicate a significant relationship between the learning rate and the accuracy attained by the model. Furthermore, there are additional investigations have been conducted on this topic, such as Rethinking Class-Balanced Methods for Long-Tailed Visual Recognition from a Domain Adaptation Perspective [13], Learning Texture Invariant Representation for Domain Adaptation of Semantic Segmentation [14], Learning Rate Schedules for Self-Organizing Maps [15], and a multitude of analogous investigations.

In Deep learning, various hyperparameters must be set to optimize the algorithm's performance, one of which is the learning rate. The choice of values for this hyperparameter is crucial because it can significantly impact the system's overall performance [16]. That is why machine learning programmers typically dedicate a considerable amount of time to fine-tuning the learning rate. Based on this issue, this research aims to determine the optimal learning rate value using the Particle Swarm Optimization (PSO) algorithm. Furthermore, this study aims to analyze the accuracy obtained based on the learning rate value obtained from PSO. Additionally, this research evaluates whether the learning rate value affects the computation time during the training phase. Several combinations of particles have been tested to expand the search area for the optimal learning rate value. This research is expected to provide insights into improving the Deep learning training process by determining the optimal learning rate value using the PSO algorithm. The results of this research can potentially help developers create more accurate and efficient models with less time and effort. This study utilizes the Recurrent Neural Network (RNN) algorithm and the GTZAN (George Tzanetakis Genre Collection) dataset [17] to evaluate the effectiveness of PSO in finding the optimal learning rate value in the case of music genre classification.

This article is structured as follows: Section 2 outlines the methodology employed to achieve the objectives outlined earlier. This section encompasses essential stages, including data preprocessing, comprehensive data analysis, feature selection, and model presentation, and it explains how particle swarm optimization can enhance the learning rate to improve model accuracy. In Section 3, we present the results obtained from testing the Recurrent Neural Network in the domain of Music Genre Classification, with a focus on the learning rate suggested by previous studies. Furthermore, this section also presents the findings of the learning rate derived from the Particle Swarm Optimization approach throughout the experimentation period. Finally, Section 4 concludes the article, offering insights into future research directions and potential areas of improvement.

2. RESEARCH METHOD

This research uses a quantitative methodology to improve the accuracy of the Recurrent Neural Network model by utilizing Particle Swarm Optimization for optimization purposes in the case of Music Genre Classification. The music genre dataset used in this study is sourced from the GTZAN public dataset. The various phases of the research process performed in this study are elucidated in a research methodology flowchart, as illustrated in Figure 1.

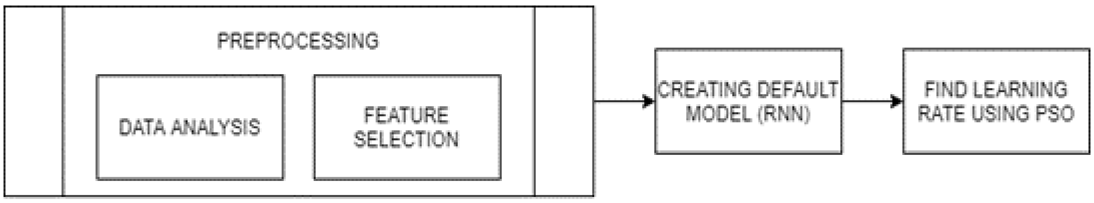

Figure 1. Research Method

2.1. Preprocessing

One of the steps involved in data mining is preprocessing, which involves transforming raw data into a more comprehensible format [18]; in this study, preprocessing was conducted in two stages: data analysis and feature selection. The data analysis stage aims to view data more simply and understandably. The GTZAN dataset [17] is a music genre dataset that has been extracted so that deep learning can directly process it. Here are some examples of data provided by the dataset in Table 1.

Table 1. An example of GTZAN data in transposed form

Feature	Data 1	Data 2	Data 3	Data 4
filename	blues.00000.wav	blues.00001.wav	blues.00002.wav	blues.00003.wav
length	661794	661794	661794	661794
chroma_stft_mean	0.35008812	0.340913594	0.363637179	0.404784709
chroma_stft_var	0.088756569	0.094980255	0.085275196	0.093999036
rms_mean	0.130227923	0.09594781	0.175570413	0.141093001
rms_var	0.002826696	0.002372739	0.002745916	0.006346346
spectral_centroid_mean	1784.16585	1530.176679	1552.811865	1070.106615
spectral_centroid_var	129774.0645	375850.0736	156467.6434	184355.9424
spectral_bandwidth_mean	2002.44906	2039.036516	1747.702312	1596.412872
spectral_bandwidth_var	85882.76132	213843.7555	76254.19226	166441.4948
rolloff_mean	3805.839606	3550.522098	3042.260232	2184.745799
rolloff_var	901505.4255	2977893.388	784034.4607	1493194.36
zero_crossing_rate_mean	0.083044821	0.056039809	0.076291207	0.033308863
zero_crossing_rate_var	0.000766946	0.001447521	0.001006829	0.000422759
harmony_mean	-4.52972E-05	0.000139581	2.10558E-06	4.58364E-07
harmony_var	0.008172282	0.005099332	0.016341973	0.019054487
perceptr_mean	7.78323E-06	-0.000177608	-1.94662E-05	-1.44832E-05
perceptr_var	0.005698182	0.003063172	0.007457626	0.002712198
tempo	123.046875	67.99958882	161.4990234	63.02400915
mfcc1_mean	-113.5706482	-207.5016937	-90.72259521	-199.5442047
mfcc1_var	2564.20752	7764.555176	3319.044922	5507.51709
mfcc2_mean	121.5717926	123.9912643	140.4463043	150.0908966
mfcc2_var	295.9138184	560.2599487	508.7650452	456.5054016
mfcc3_mean	-19.16814232	8.955126762	-29.09388924	5.662678242
mfcc3_var	235.5744324	572.8109131	411.7812195	257.1611633
mfcc4_mean	42.36642075	35.8776474	31.6843338	26.85907936
mfcc4_var	151.1068726	264.5061035	144.0903168	158.2673035
mfcc5_mean	-6.364664078	2.907319784	-13.98450375	1.77139926
mfcc5_var	167.9347992	279.9329224	155.4937592	268.0343933
mfcc6_mean	18.62349892	21.51046562	25.7647419	14.23403072
mfcc6_var	89.18083954	156.4770966	74.54840088	126.7941284

(Continued on the next page)

Table 1 (Continued)

Feature	Data 1	Data 2	Data 3	Data 4
mfcc7_mean	-13.7048912	-8.560436249	-13.66487503	-4.832006454
mfcc7_var	67.66049194	200.8491821	106.9818268	155.9120789
mfcc8_mean	15.34315014	23.37068558	11.63993359	9.286494255
mfcc8_var	68.93257904	142.555954	106.5748749	81.27374268
mfcc9_mean	-12.27410984	-10.09966087	-11.78364277	-0.759186447
mfcc9_var	82.20420074	166.1085205	65.44794464	92.11408997
mfcc10_mean	10.97657204	11.90049744	9.71876049	8.137606621
mfcc10_var	63.38631058	104.3586121	67.90885925	71.31407928
mfcc11_mean	-8.326573372	-5.55563879	-13.13380337	-3.200653315
mfcc11_var	61.77309418	105.1736298	57.78142548	110.2366867
mfcc12_mean	8.803792	5.376327038	5.791199207	6.079319
mfcc12_var	51.24412537	96.19721222	64.48020935	48.2519989
mfcc13_mean	-3.6723001	-2.231760263	-8.907628059	-2.480173826
mfcc13_var	41.21741486	64.91429138	60.38515091	56.79940033
mfcc14_mean	5.7479949	4.220139503	-1.077000499	-1.079305053
mfcc14_var	40.55447769	73.15253448	57.71113586	62.28990173
mfcc15_mean	-5.162881851	-6.01214838	-9.229273796	-2.870788574
mfcc15_var	49.77542114	52.42214203	36.58098602	51.65159225
mfcc16_mean	0.752740204	0.927997768	2.451689959	0.780873835
mfcc16_var	52.42090988	55.35640335	40.59876633	44.42775345
mfcc17_mean	-1.690214634	-0.73112458	-7.729092598	-3.319596529
mfcc17_var	36.52407074	60.31452942	47.63942719	50.20667267
mfcc18_mean	-0.408979177	0.295072943	-1.816406965	0.636965036
mfcc18_var	41.59710312	48.12059784	52.38214111	37.31912994
mfcc19_mean	-2.303522587	-0.283518016	-3.439720392	-0.619121194
mfcc19_var	55.06292343	51.10618973	46.63965988	37.25973892
mfcc20_mean	1.221290708	0.531216502	-2.231258392	-3.407448292
mfcc20_var	46.93603516	45.78628159	30.57302475	31.94933891
label	blues	blues	blues	blues

The data shown in Table 1 above is a small portion of the total data provided by the GTZAN dataset. The GTZAN dataset's total data includes 1000 rows of data, 58 features, 1 ID, 1 label, and 10 classes. The distribution of data for each class is in Figure 2.

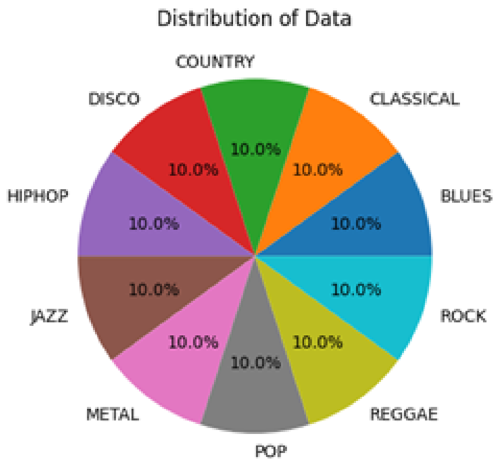

Figure 2. Distribution of Data

There are 1000 rows of data and 10 classes in the dataset. According to Figure 2, each class has an equal data distribution of 10%, meaning there are 100 rows of data for each class. Based on this information, this research performs a binary classification of music genres for each class in the datasetfeature Selection. The GTZAN dataset provides a variety of 58 features that can be used in deep learning, and a number of these features show significant correlations with each other. This relationship is highlighted in Figure 3.

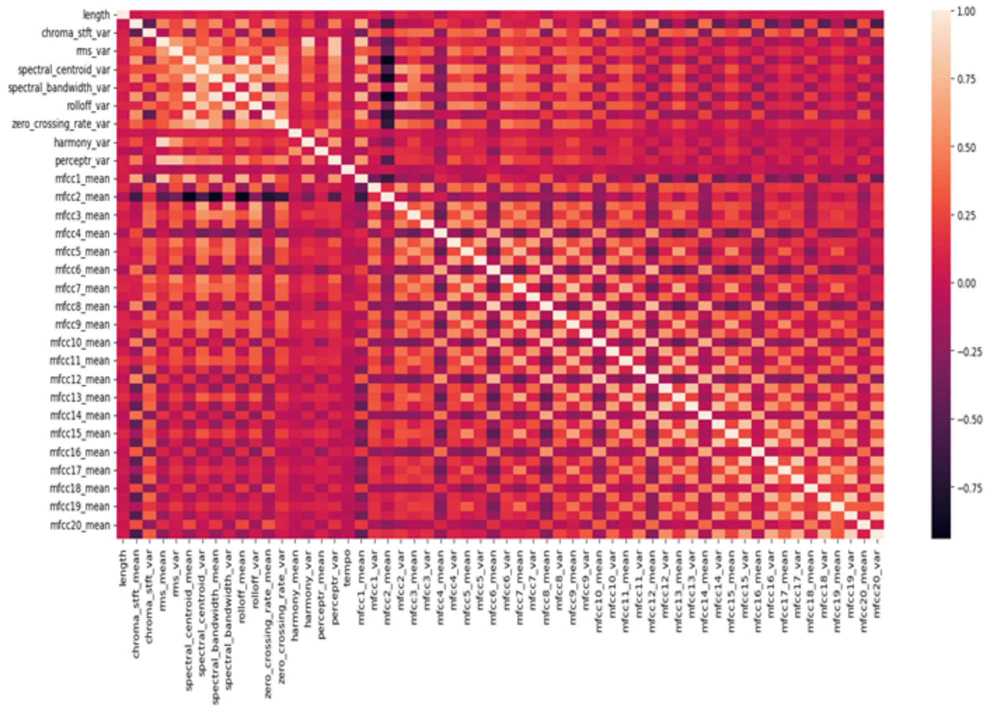

Figure 3. Matrix of correlations before removing correlated features

Based on Figure 3 above, more than 50% of the features have a very high correlation coefficient between one another. Therefore, all features with a coefficient greater than 0.6 have been removed to avoid multicollinearity, leaving only 16 features, as shown in Figure 4.

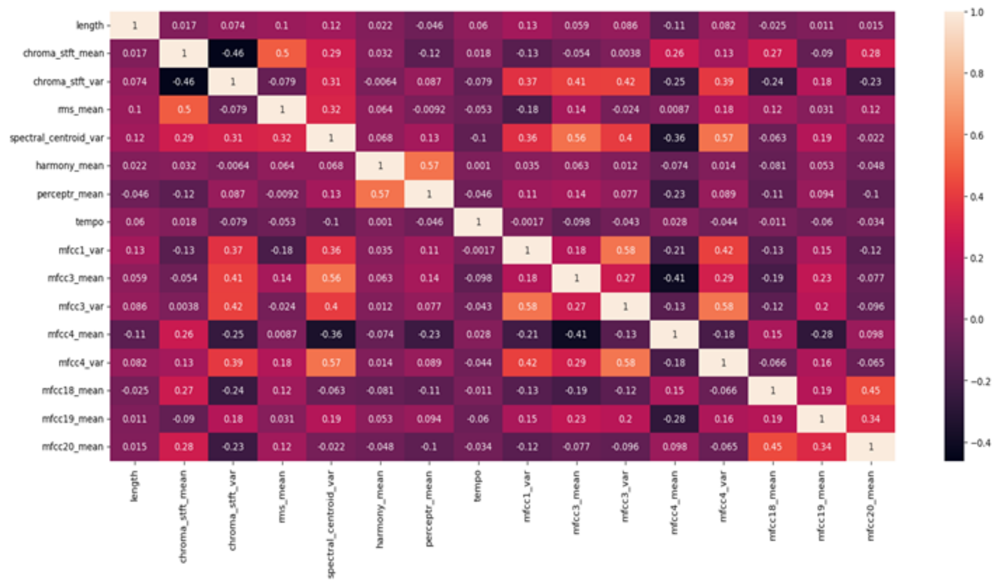

Figure 4. Matrix of correlations before removing correlated features

After conducting feature analysis and selecting appropriate data, his research develops an RNN model to classify the data based on the selected features.

2.2. Creating Default Model (RNN)

One of the most widely used models for analyzing sequential data is Recurrent Neural Networks (RNNs). These networks apply the same operation to each input token in sequence. RNNs are built with a recurrent structure that enables them to identify dependencies among different sequences' tokens. This quality has been effective in numerous applications, such as natural language processing and speech recognition [19]. This research uses a Recurrent Neural Network (RNN) architecture, as shown in Figure 5.

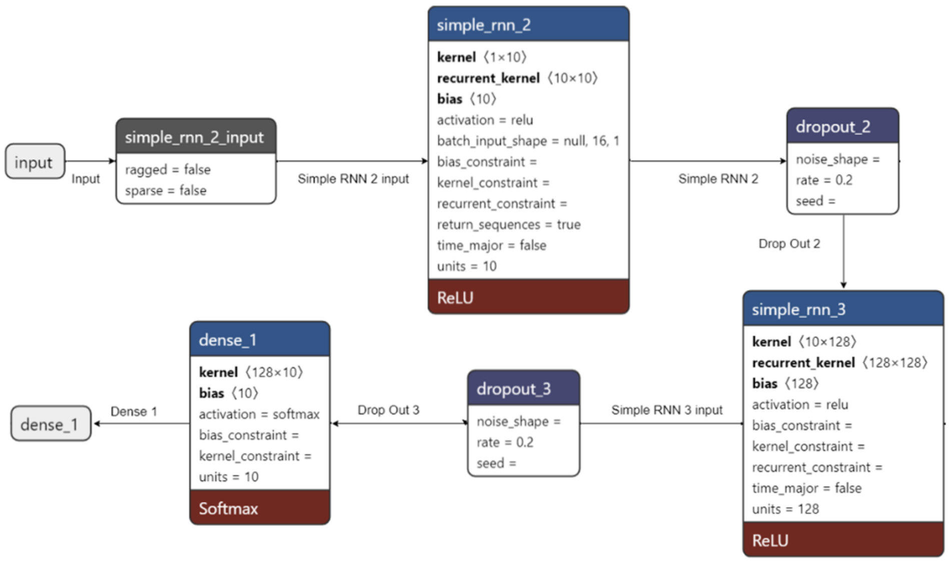

Figure 5. The Recurrent Neural Network (RNN) architecture for binary music genre classification

The model consists of three layers. The first layer is a Simple RNN layer with 10 units, using the relu activation function. This layer returns a sequence of outputs, as the parameter return_sequences is set to True. The second layer is a Dropout layer with a rate of 0.2, randomly dropping out a certain percentage of the input units during training to prevent overfitting. The third layer is another Simple RNN layer with 128 units and a ReLU activation function, followed by another Dropout layer with a rate of 0.2. The final layer is a Dense layer with 10 units and a softmax activation function used for multi-class classification problems. Overall, the model uses two Simple RNN layers with dropout regularization to avoid overfitting, followed by a dense output layer for classification.

2.3. Find Learning Rate Using PSO

Particle Swarm Optimization (PSO) algorithm is a population-based optimization technique inspired by the social behavior of fish schooling or bird flocking; in other words, Particle swarm optimization (PSO) is a stochastic population-based optimization algorithm inspired by the interactions of individuals in a social world [20], introduced by Kennedy and Eberhart. It has similarities to transformative computation methods such as Genetic Algorithm (GA). PSO begins with a group of irregular arrangements and refreshes generations to look for optima. Unlike GA, PSO doesn't have evolution administrators such as mutation and crossover. The algorithm's potential solutions, referred to as particles, move through the problem space by following the current optimal particles [21]. A diagram illustrating the PSO methodology is presented in Figure 6.

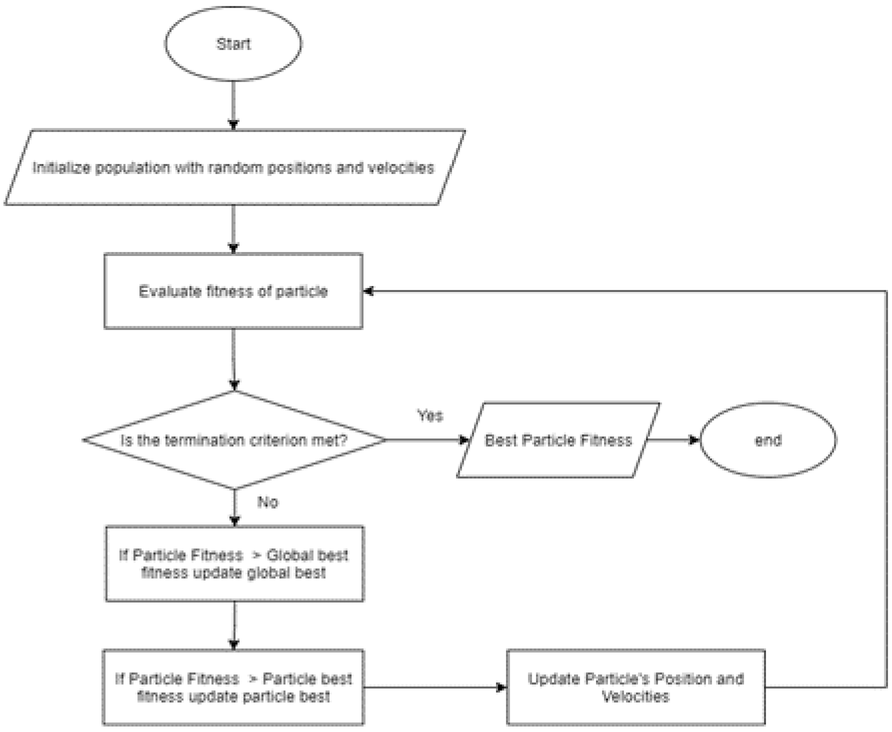
Start
Initialize population with random positions and velocities
Evaluate fitness of particle
Is the termination criterion met?
Yes
Best Particle Fitness
end
No
If Particle Fitness > Global best fitness update global best
If Particle Fitness > Particle best fitness update particle best
Update Particle's Position and Velocities

Figure 6. Methodology of Particle Swarm Optimization (PSO)

Based on the methodology depicted in Figure 6, this study optimized the learning rate using the same methodology but modified it to be as indicated in Figure 7.

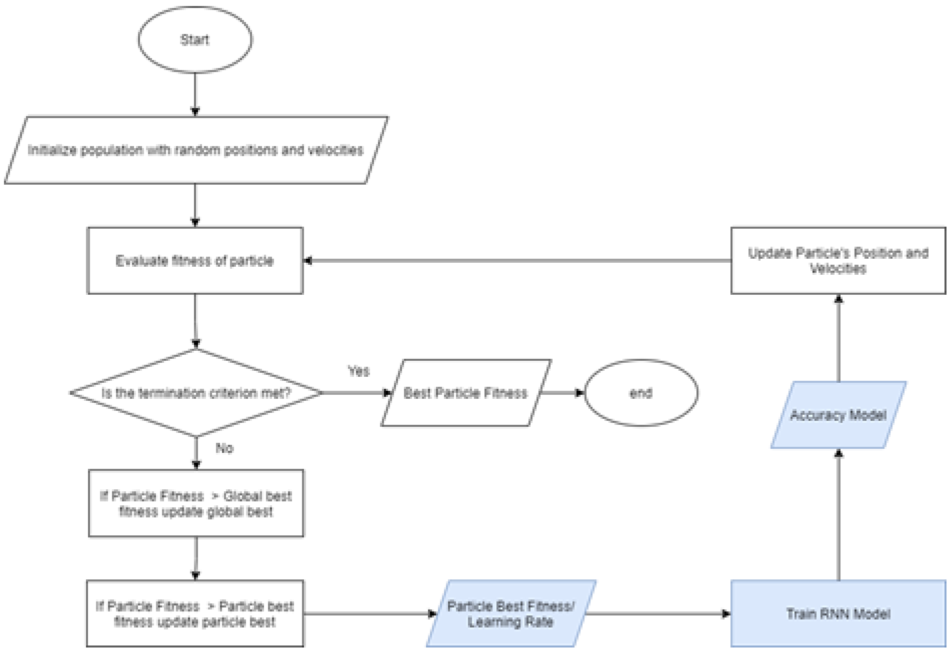
Start
Initialize population with random positions and velocities
Evaluate fitness of particle
Update Particle's Position and Velocities
Is the termination criterion met?
Yes
Best Particle Fitness
end
Accuracy Model
No
If Particle Fitness > Global best fitness update global best
If Particle Fitness > Particle best fitness update particle best
Particle Best Fitness/ Learning Rate
Train RNN Model

Figure 7. The PSO methodology for determining the RNN Learning Rate

Figure 7 depicts the application of the PSO algorithm to the RNN algorithm to search for the appropriate learning rate. In the scenario illustrated in the figure, the PSO algorithm halts only when one of the particles satisfies the criteria or discovers the optimal learning rate for the RNN model.

3. RESULT AND ANALYSIS

At this stage, we provide an overview of the analysis and results obtained in our research, which have been categorized into the following points.

3.1. Effect of Default Learning Rate on RNN Performance

As discussed in the previous sub-chapter, this study utilizes the Recurrent Neural Network model and the GTZAN dataset to evaluate model performance against learning rates determined by the Particle Swarm Optimization algorithm. However, before delving deeper into that topic, this study presents the performance results of the RNN model when classifying the GTZAN dataset using the default learning rate value of 0.01 [9]. The results obtained from our experiment are presented in Figure 8.

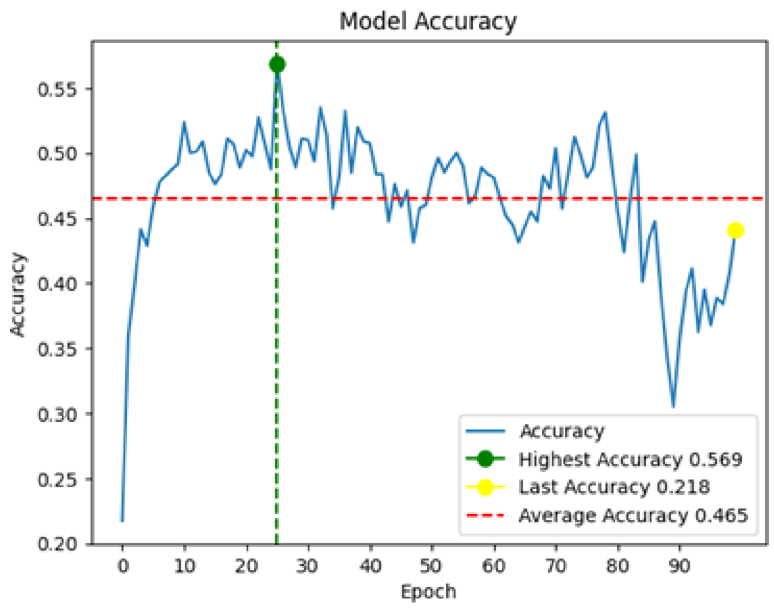

Figure 8. RNN model accuracy in GTZAN dataset classification with default learning rate

Figure 8 indicates that the highest accuracy value of 0.569 was achieved at epoch 25, but the accuracy value gradually decreased after that point. The graph also shows that the accuracy is unstable, resulting in an average accuracy of only 0.465. These results were produced by a model that ran 32 batches for 100 epochs. Ultimately, the final accuracy result is 0.218.

3.2. Comparing RNN Model Accuracy with Various Learning Rates in Prior Studies

Instead of using default values, previous research used learning rate values of 0.0001 [10], 0.001, 0.003, 0.01, 0.03, and 0.1 [11] for each model they used. Before attempting to use the learning rate generated by the PSO algorithm, this study also attempts to use the learning rate values from previous research to test whether they can produce better accuracy than the learning rate values generated by the PSO algorithm. The results of the experiment are presented in Figure 9.

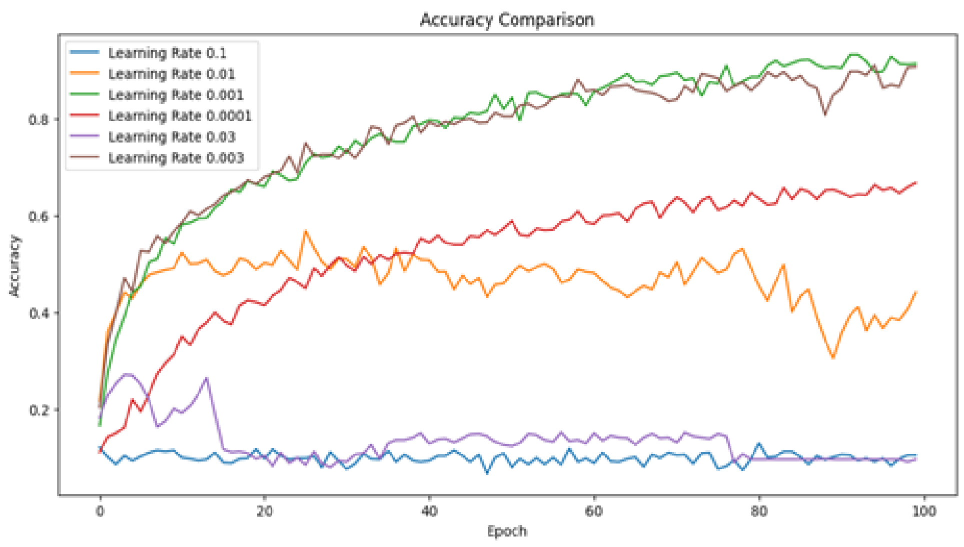

Figure 9. RNN Model Accuracy with Previous Research Learning Rates

Based on the results shown in Figure 9, the RNN model achieved the highest accuracy of 0.933 in epoch 92 using a learning rate of 0.001. The average accuracy for this learning rate across epochs was 0.773, with the final accuracy reaching 0.914. It is worth noting that a learning rate of 0.003 also performed well, reaching the highest accuracy of 0.911 in epoch 95. The average accuracy for this learning rate was 0.768, with the final accuracy being 0.908. Despite a relatively low final accuracy value of 0.668, the learning rate of 0.0001 showed a positive trend in its graph. The computation time for this model was 62.785 seconds. In contrast, as the learning rates increased to 0.01, 0.03, and 0.1, the accuracies noticeably decreased, reaching values of 0.442, 0.096, and 0.105, respectively. Additionally, the computation times for these models were 67.628 seconds, 86.428 seconds, and 78.893 seconds, respectively.

3.3. Impact of Learning Rate Produce by PSO on RNN Model Accuracy

This study optimizes the learning rate using Particle Swarm Optimization on an RNN model with a search range starting from 0.001 as the lower limit and 0.002 as the upper limit. Additionally, the study explores optimization by considering various combinations in the search particles. The results of the PSO optimization are presented in the following Table 2.

Table 2. Impact of Learning Rate Produced by PSO on RNN Model Accuracy Results

Experiment	Learning Rate	Highest Accuracy	Epoch	Average Accuracy	Ultimate Accuracy	Computation Time (s)	Particle	Time Optimization PSO (s)
1	0.00100928	0.940	90	0.775	0.923	82.925	15	8481.20
2	0.001019	0.935	97	0.780	0.909	55.960	15	3865.10
3	0.00123613	0.944	100	0.793	0.944	115.591	15	13632.60
4	0.00124751	0.959	90	0.801	0.945	86.109	10	2271.70
5	0.0013035	0.951	97	0.792	0.929	63.715	10	12936.32
6	0.00146656	0.955	100	0.806	0.955	104.954	5	2457.40
7	0.00149967	0.944	98	0.799	0.934	67.584	5	3165.70
8	0.00151369	0.946	96	0.807	0.929	76.510	5	1138.10
9	0.00157388	0.941	96	0.792	0.935	65.539	5	4364.60
10	0.00163064	0.957	98	0.795	0.942	74.940	10	5728.10
11	0.00177725	0.933	94	0.791	0.919	91.416	15	27923.10
12	0.00185998	0.940	89	0.812	0.926	58.596	10	16170.40

It is evident that a learning rate of 0.00146656 achieves the highest accuracy among the tested rates, with a value of 0.955. This learning rate also demonstrates a respectable average accuracy of 0.806 and an ultimate accuracy of 0.955. It accomplishes these results within 100 epochs and has a computation time of 104.954 seconds (about 1 minute 45 seconds). Notably, this performance is obtained with a particle size of 5 and a PSO time optimization of 2457.40, as shown in Figure 10.

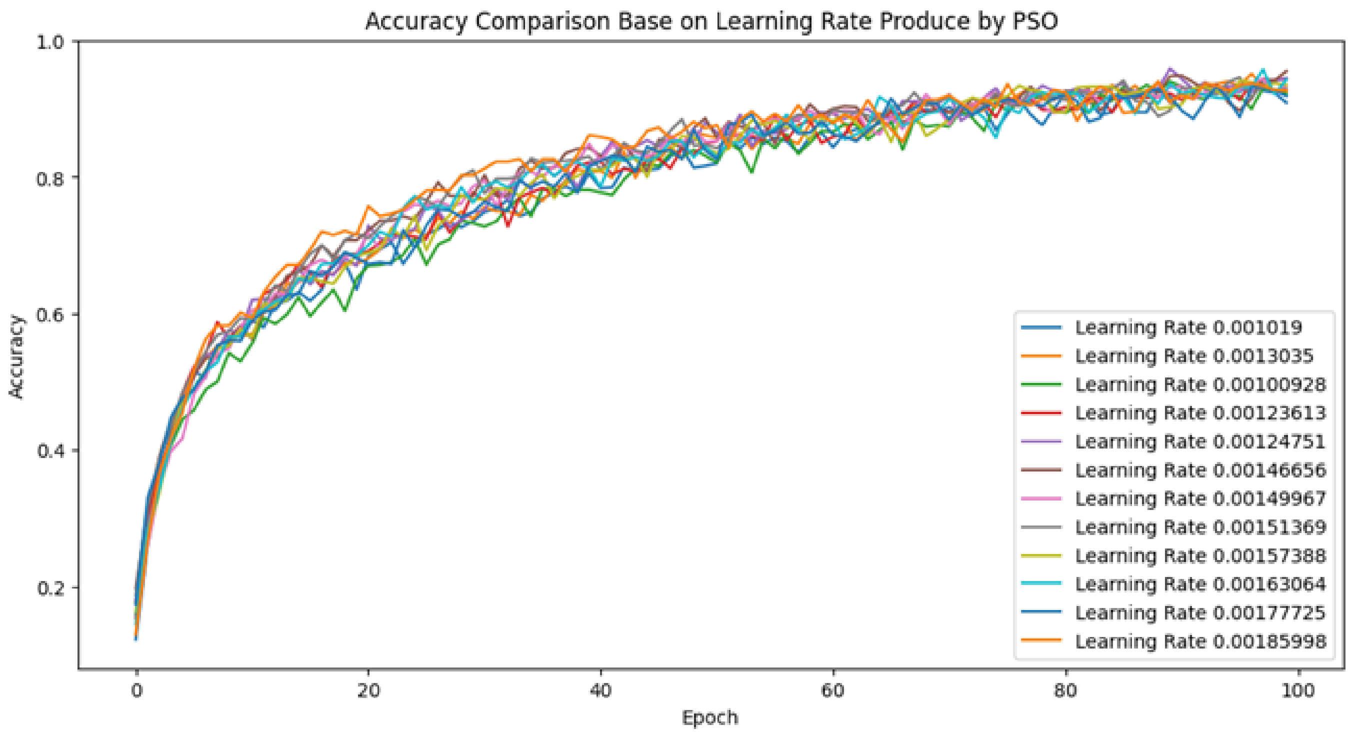

Figure 10. RNN Model Accuracy Using Learning Rate Produce by PSO

High accuracy is likewise produced with a learning rate of 0.001574, with a maximum accuracy of 0.952, an average accuracy of 0.796, and a final accuracy of 0.945. These results are achieved within 94 epochs and take 63.7 seconds to compute. The particle size for this learning rate is 5, and the corresponding PSO time optimization is 4364.60. Another noteworthy learning rate is 0.001631, which exhibits a competitive highest accuracy of 0.951, an average accuracy of 0.807, and an ultimate accuracy of 0.946. These results are achieved within 97 epochs and a computation time of 60.9 seconds. The particle size for this learning rate is 5, and the corresponding PSO time optimization is 5728.10. Considering both accuracy and computation time, the learning rate of 0.00146656 stands out as the optimal choice. It achieves a high accuracy level of 0.955 within a relatively shorter computation time of 104.954 seconds (about 1 minute 45 seconds). Furthermore, it demonstrates a reasonably efficient PSO time optimization of 2457.40. Therefore, utilizing a learning rate of 0.00146656 with PSO time optimization can likely yield favorable results for the RNN model.

4. CONCLUSION

As a result, this study found that initially, testing using a learning rate of 0.01 gradually decreased accuracy after going through the 25th epoch and ended at an accuracy of 0.218 at the 100th epoch. An extensive exploration of various learning rates obtained from previous research (0.0001, 0.001, 0.003, 0.01, 0.03, and 0.1) was conducted to enhance the model's accuracy. Among these, the learning rate of 0.001 emerged as the most promising, exhibiting a peak accuracy of 0.933 during epoch 92. This learning rate showcased an impressive average accuracy of 0.773 and a final accuracy of 0.914. The learning rate of 0.003 also demonstrated notable performance, reaching a pinnacle accuracy of 0.911. Particle Swarm Optimization (PSO) was applied to the RNN model to further optimize the learning rate. The PSO algorithm explored various learning rate values within a search range of 0.001 to 0.002, with different particle sizes and optimization times. The results showed that a learning rate of 0.00146656 achieved the highest accuracy of 0.955, with an average accuracy of 0.806 and an ultimate accuracy of 0.955. This learning rate was performed within 100 epochs, taking 104.954 seconds (about 1 minute 45 seconds) to compute. Another notable learning rate was 0.001574, which achieved a maximum accuracy of 0.952, an average accuracy of 0.796, and a final accuracy of 0.945 within 94 epochs and 63.7 seconds of computation time. A learning rate of 0.001631 also demonstrated a competitive highest accuracy of 0.951. Considering both accuracy and computation time, the learning rate of 0.00146656 emerged as the optimal choice. It achieved a high accuracy level of 0.955 within a relatively shorter computation time of 104.954 seconds and a reasonably efficient PSO time optimization of 2457.40. Therefore, utilizing a learning rate of 0.00146656 with PSO time optimization holds the potential to yield favorable results for the RNN model, surpassing the performance of both the default learning rate and the learning rates from previous research. Future research recommendations entail conducting further investigations within the same domain, encompassing comparative analyses of diverse optimization algorithms to ascertain the superior algorithm in generating optimal learning rates. Additionally, comparing the computational time required to produce these learning rates would be valuable.

5. ACKNOWLEDGEMENTS

The authors express their sincere gratitude to all individuals who contributed to this work, with special appreciation extended to the anonymous reviewers, the chief/managing editors, and the esteemed staff of Matrik: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer.

6. DECLARATIONS

AUTHOR CONTIBUTION

Three authors conducted This research collaboratively, each contributing to different aspects of the project. Muhammad Rizki as Author 1, conceptualized and designed the study and collected, analyzed, and interpreted the data. Arief Hermawan, as Author 2, was responsible for the preparation of the manuscript. Lastly, Donny Avianto, as Author 3, carried out specific tasks within the research project. The authors would like to acknowledge their individual contributions to the completion of this study.

FUNDING STATEMENT

This research did not receive dedicated funding from any public, commercial, or not-for-profit organization or agency.

COMPETING INTEREST

We declare that I have no financial, general, or institutional competing interests to disclose.

REFERENCES

[1] J. Yang, Y. Li, Q. Liu, L. Li, A. Feng, T. Wang, S. Zheng, A. Xu, and J. Lyu, "Brief introduction of medical database and data mining technology in big data era," *Journal of Evidence-Based Medicine*, vol. 13, no. 1, pp. 57–69, 2020, https://doi.org/10.1111/jebm.12373.

[2] A. Lewkowycz, Y. Bahri, E. Dyer, J. Sohl-Dickstein, and G. Gur-Ari, "The large learning rate phase of deep learning: the catapult mechanism," *arXiv:2003.02218v1*, no. 4 Mar, 2020.

[3] Y. Ming, X. Meng, C. Fan, and H. Yu, "Deep learning for monocular depth estimation: A review," *Neurocomputing*, vol. 438, pp. 14–33, 2021, https://doi.org/10.1016/j.neucom.2020.12.089.

[4] S. Xie, Z. Yu, and Z. Lv, "Multi-disease prediction based on deep learning: A survey," *CMES - Computer Modeling in Engineering and Sciences*, vol. 127, no. 3, 2021, https://doi.org/10.32604/CMES.2021.016728.

[5] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep Learning-based Text Classification: A Comprehensive Review," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–40, apr 2022, https://doi.org/10.1145/3439726.

[6] M. R. Karim, O. Beyan, A. Zappa, I. G. Costa, D. Rebholz-Schuhmann, M. Cochez, and S. Decker, "Deep learning-based clustering approaches for bioinformatics," *Briefings in Bioinformatics*, vol. 22, no. 1, pp. 393–415, 2021, https://doi.org/10.1093/bib/bbz170.

[7] J. Konar, P. Khandelwal, and R. Tripathi, "Comparison of Various Learning Rate Scheduling Techniques on Convolutional Neural Network," *2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science, SCEECS 2020*, 2020, https://doi.org/10.1109/SCEECS48394.2020.94.

[8] M. Nagy, M. El-Sersy, M. A. Mohamed, and K. S. Alou, “A Modied Method for Detecting DDoS Attacks Based on Articial Neural Networks,” *The 1st International Conference on Information Technology (IEEE/ITMUSTCONF)*, 2019.

[9] A. R. Asif, A. Waris, S. O. Gilani, M. Jamil, H. Ashraf, M. Shafique, and I. K. Niazi, “Performance evaluation of convolutional neural network for hand gesture recognition using EMG,” *Sensors (Switzerland)*, vol. 20, no. 6, pp. 1–11, 2020, https://doi.org/10.3390/s20061642.

[10] J. Jepkoech, D. M. Mugo, B. K. Kenduiywo, and E. C. Too, “The Effect of Adaptive Learning Rate on the Accuracy of Neural Networks,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 8, pp. 736–751, 2021, https://doi.org/10.14569/IJACSA.2021.0120885.

[11] R. Guha, “Benchmarking Gradient Based Optimizers’ Sensitivity to Learning Rate,” *SSRN Electronic Journal*, no. January, pp. 1–33, 2023.

[12] Y. Aprizal, R. I. Zainal, and A. Afriyudi, “Perbandingan Metode Backpropagation dan Learning Vector Quantization (LVQ) Dalam Menggali Potensi Mahasiswa Baru di STMIK PalComTech,” *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 18, no. 2, pp. 294–301, 2019, https://doi.org/10.30812/matrik.v18i2.387.

[13] M. A. Jamal, M. Brown, M. H. Yang, L. Wang, and B. Gong, “Rethinking Class-Balanced Methods for Long-Tailed Visual Recognition from a Domain Adaptation Perspective,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7607–7616, 2020, https://doi.org/10.1109/CVPR42600.2020.00763.

[14] M. Kim and H. Byun, “Learning Texture Invariant Representation for Domain Adaptation of Semantic Segmentation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 12 972–12 981, 2020, https://doi.org/10.1109/CVPR42600.2020.01299.

[15] F. Mulier and V. Cherkassky, “Learning rate schedules for self-organizing maps,” *Proceedings - International Conference on Pattern Recognition*, vol. 2, pp. 224–228, 1994, https://doi.org/10.1109/icpr.1994.576908.

[16] B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A. L. Boulesteix, D. Deng, and M. Lindauer, “Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 13, no. 2, pp. 1–43, 2023, https://doi.org/10.1002/widm.1484.

[17] ANDRADA, “GTZAN Dataset - Music Genre Classification,” 2020.

[18] A. Anggrawan, “Application of KNN Machine Learning and Fuzzy C-Means to Diagnose Diabetes,” vol. 22, no. 2, pp. 405–418, 2023, https://doi.org/10.30812/matrik.v22i2.2777.

[19] Z. Allen-Zhu and Y. Li, “Can SGD learn recurrent neural networks with provable generalization?” *Advances in Neural Information Processing Systems*, vol. 32, no. NeurIPS, 2019.

[20] M. Jahandideh-Tehrani, O. Bozorg-Haddad, and H. A. Loáiciga, “Application of particle swarm optimization to water management: an introduction and overview,” *Environmental Monitoring and Assessment*, vol. 192, no. 5, 2020, https://doi.org/10.1007/s10661-020-8228-z.

[21] T. H. Zhao, M. I. Khan, and Y. M. Chu, “Artificial neural networking (ANN) analysis for heat and entropy generation in flow of non-Newtonian fluid between two rotating disks,” *Mathematical Methods in the Applied Sciences*, vol. 46, no. 3, pp. 3012–3030, 2023, https://doi.org/10.1002/mma.7310.