









Category: STEM (Science, Technology, Engineering and Mathematics)

ORIGINAL

Advanced Dual-Optimized Neural Network Model with Integrated CSO and OBD for Precise Classification and Prediction of North Indian Light Classical Music Genres

Modelo avanzado de red neuronal de doble optimización con CSO y DAB integrados para la clasificación y predicción precisas de los géneros de música clásica ligera del norte de la India

Pavani G¹ , Satishkumar Patnala² , Sangita Chakraborty³ , Divvela Surendra⁴ , Katakam Ranga Narayana⁵ , Jyothi N M⁶ 

¹Velagapudi Ramakrishna Siddhartha Engineering College, Siddhartha Academy of Higher Education, Department of English. Kannur, India.

²Anil Neerukonda Institute of Technology & Sciences(A), Department of IT. Visakhapatnam, India.

³Gopal Narayan Singh University, Department FIT. Jamuhar, India.

⁴Koneru Lakshmaiah Education Foundation, Department of English. Vaddeswaram, India.

⁵GITAM University, Department of Information Technology. Visakhapatnam, India.

⁶Koneru Lakshmaiah Education Foundation, Computer Science Engineering. Vaddeswaram, India.

Cite as: Pavani G, Satishkumar P, Chakraborty S, Surendra D, Ranga Narayana K, Jyothi NM. Advanced Dual-Optimized Neural Network Model with Integrated CSO and OBD for Precise Classification and Prediction of North Indian Light Classical Music Genres. Salud, Ciencia y Tecnología - Serie de Conferencias. 2024; 3:805. <https://doi.org/10.56294/sctconf2024805>

Submitted: 11-01-2024

Revised: 13-04-2024

Accepted: 15-07-2024

Published: 16-07-2024

Editor: Dr. William Castillo-González 

ABSTRACT

Introduction: categorizing North Indian Light Classical Music genres presents a considerable challenge due to their intricate nature. This research introduces a Dual Optimized Neural Network (DONN) model designed to achieve elevated levels of accuracy and efficiency, thereby enhancing the understanding of these music genres. Creating a network of artificial neural with accurate classification and prediction of given genres is the primary objective. This is achieved through the integration of Cat Swarm Optimization (CSO) for enhanced adaptability and Optimal Brain Damage (OBD) for effective network pruning.

Methods: the DONN model employs CSO to investigate the solution space effectively while using OBD to minimize unnecessary network connections, thereby improving both computational efficiency and generalization capabilities. The methodology involves modelling the network using a dataset of North Indian Light Classical Music, optimizing the search process with CSO, and applying OBD for network pruning.

Results: the DONN model demonstrated a remarkable 98 % accuracy in classifying eleven distinct genres, outperforming previous methods, and highlighting superior classification accuracy and resilience. Compared to earlier research work and Swarm Optimization like Bat and Ant Colony, and Particle Swarm Algorithm, this model shows higher accuracy and efficiency. The fusion of CSO and OBD significantly enhances performance, improving generalization and reducing computational complexity.

Conclusions: overall, the DONN model, optimized with CSO and OBD, significantly advances the classification and prediction of North Indian Light Classical Music genres. This research offers a robust and reliable tool for music classification, contributing to a deeper understanding and appreciation of these genres.

Keywords: Artificial Neural Network; Cat Swarm Optimization; Double Optimized Neural Network; Optimal Brain Damage; Music Genres.

RESUMEN

Introducción: la categorización de los géneros de Música Clásica Ligera del Norte de la India presenta un reto considerable debido a su naturaleza intrincada. Esta investigación introduce un modelo de Red Neuronal

Dual Optimizada (DONN) diseñado para alcanzar elevados niveles de precisión y eficiencia, mejorando así la comprensión de estos géneros musicales. El objetivo principal es crear una red neuronal artificial con una clasificación y predicción precisas de determinados géneros. Esto se consigue mediante la integración de la Optimización de Enjambre de Gatos (CSO) para una mayor adaptabilidad y el Daño Cerebral Óptimo (DCO) para una poda eficaz de la red.

Métodos: el modelo DONN emplea CSO para investigar el espacio de soluciones de forma efectiva mientras que utiliza OBD para minimizar las conexiones de red innecesarias, mejorando así tanto la eficiencia computacional como la capacidad de generalización. La metodología consiste en modelar la red utilizando un conjunto de datos de música clásica ligera del norte de la India, optimizar el proceso de búsqueda con CSO y aplicar OBD para la poda de la red.

Resultados: el modelo DONN demostró una notable precisión del 98 % en la clasificación de once géneros distintos, superando a métodos anteriores y destacando una precisión de clasificación y una resistencia superiores. En comparación con anteriores trabajos de investigación y optimización de enjambres como Bat y Ant Colony, y Particle Swarm Algorithm, este modelo muestra una mayor precisión y eficiencia. La fusión de CSO y DAB aumenta significativamente el rendimiento, mejorando la generalización y reduciendo la complejidad computacional.

Conclusiones: en general, el modelo DONN, optimizado con CSO y OBD, avanza significativamente en la clasificación y predicción de los géneros de Música Clásica Ligera del Norte de la India. Esta investigación ofrece una herramienta robusta y fiable para la clasificación musical, contribuyendo a una comprensión y apreciación más profundas de estos géneros.

Palabras clave: Red Neuronal Artificial; Optimización por Enjambre de Gatos; Red Neuronal Doblemente Optimizada; Daño Cerebral Óptimo; Géneros Musicales.

INTRODUCTION

India is a country of various cultures, everyone having its own genre of music since ancient times. Classical music of North India can be broadly classified into Carnatic, Hindustani, and light classical or semi-classical music. Examples of North Indian semi-classical music includes Chaiti, Naty Sangeet, Bhajan, Qawwali, Keertan, Thumri, Kajri, Tappa, Dadra, Dhun, and Ghazal. These genres, though similar, have subtle differences and are generally softer in style. This research aims to classify North Indian semi-classical music using advanced optimization techniques to address the challenge posed by the thin decision boundaries among various classes. The primary goal of this study is to develop a robust classification system that accurately identifies and categorizes North Indian semi-classical music genres, facilitating quicker searches and enhancing understanding of music industry and enthusiasts. The classification task is intricate due to the subtle distinctions between these musical forms, making it difficult for common people to identify these genres due to their diverse origins. Additionally, the research seeks to predict precise class labels for new additions to the repertoire, contributing to a comprehensive and accessible musical database.

To achieve this, we introduce a Dual Optimized Neural Network (DONN) model. The model integrates Cat Swarm Optimization (CSO) for effective exploration and adaptability, and Optimal Brain Damage (OBD) for pruning insignificant connections in the Artificial Neural Network (ANN) architecture. This double-folded optimization approach enhances both the efficiency and generalization capabilities of the model, resulting in high performance for multi-class classification of semi-classical music genres, with eleven different class labels. The methodology involves training a fully connected feed-forward neural network using a dataset of North Indian semi-classical music. CSO is employed to optimize the search process, while OBD is applied to prune the network and reduce unnecessary connections.

The model's performance in contrast to alternative swarm optimization methods, including Particle, Bat and Ant colony Swarm Algorithm is carried out. This approach ensures that the model achieves high accuracy and resilience in classifying the music genres, overcoming the challenges posed by the thin decision boundaries among various classes. The extent of the research covers obtaining audio attributes of the music dataset, inputting them into the ANN model, applying Optimization of Cat Swarm and Optimal Brain Damage, and observing the model's performance in music genre classification. The remaining part of the paperwork is structured into sections on the literature survey, pre-processing and architecture of the model, proposed methodology, experimental setup, results, discussion, and conclusion.

Literature Survey is carried out on recent years. A music classification system is developed by combining one-dimensional convolution of a recurrent neural network with single-dimensional convolution and a bidirectional recurrent neural network and adjusting output weights to improve representation of music style features. Its effectiveness was evaluated through comparison and experiments on the GTZAN dataset and got better results in comparison with the traditional methods.⁽¹⁾ The sonogram dataset reached 97,6 % accuracy with Google

Net for music recognition, while a parallel Google Net method converted acoustic features to speech in real-time, improving training efficiency and output quality. Additionally, Resnet18 and Shuffle net achieved 97,2 % accuracy when trained on spectrogram data, confirming the experiment's reliability.⁽²⁾ The proposed system, comprising data preparation, feature extraction, and genre classification, employs a new neural network, validated through experiments on various datasets, including GTZAN and Indian Music Genre, demonstrating its effectiveness compared to existing methods and achieved good results.⁽³⁾ Using a pretrained model trained on a large dataset and fine-tuning its weights is a common method to adapt it for new tasks, which involves employing frame acquisition and confusion matrix for output derivation. The challenge lies in categorizing audio files by genre, essential for applications like Saavan, Wynk, and Spotify, to automatically tag music frames and improve user satisfaction.⁽⁴⁾

Utilizing the Pitch2vec approach as a preprocessing step, this technique transforms pitches from MIDI files into vector sequences. By incorporating deep learning techniques, it achieves an accuracy of 95,87 %.⁽⁵⁾ Relying on the acoustic attributes of music, one method introduces a novel deep neural network model. This model is utilized in both a music genre classification system and a recommendation engine, extracting representative features for both tasks from the same dataset.⁽⁶⁾ Exploration of the feature sequence mechanism to enhance music design feedback incorporates diverse music sequence metrics. This entails measuring type probabilities via SoftMax activation, computing cross-loss function values, and applying the Adam optimization algorithm for network model refinement, followed by the development of an independent adaptive learning rate scheme.⁽⁷⁾ The proposed method preprocesses input signals and describes their characteristics using Mel Frequency Cepstral Coefficients (MFCC) and Short-Time Fourier Transform (STFT) features. Then, two separate convolutional neural network (CNN) models are employed to analyze MFCC and STFT data, resulting in a classification accuracy of 95,2 % according to experimental results.⁽⁸⁾

In the past ten years, music-streaming services have experienced significant growth, with Pandora being a pioneering force in popularizing streaming music through its successful deployment of genre-based music analysis, which relies on human-annotated content.⁽⁹⁾ The system utilizes spectrogram feature values from song slices as input for a CNN to classify songs by genre and implements a recommendation system based on user preferences. Extensive experiments on the GTZAN dataset demonstrate the system's effectiveness compared to other methods.⁽¹⁰⁾ This research investigates how well different features predict music genre using machine learning. It examines features such as Mel-frequency cepstral coefficients, key features from GTZAN, and human-understandable features from Spotify, highlighting a balance between accuracy and interpretability.⁽¹¹⁾ This study compares the performance of two model classes: one employing a deep learning CNN model trained on audio spectrograms to predict genre labels, and another utilizing hand-crafted features from both time and frequency domains. Applying four machine learning models on the Audio dataset, on ensemble classifier accuracy of 0,894 is obtained.⁽¹²⁾

Comprising three primary stages—data readiness, feature mining, and categorization—the proposed system employs a novel neural network for music genre classification. By leveraging spectrograph features from short song clips, the CNN architecture incorporates multi-scale time-frequency information, resulting in classification accuracies of 93,9 %, 96,7 %, and 97,2 % on GTZAN, Ballroom, and Extended Ballroom benchmark datasets, respectively.^(13,14,15) Training a Convolutional Neural Networks model on Mel-Frequency Cepstral Coefficients achieved an accuracy of 43 %. Additionally, a novel texture selector based on K-Means was introduced to identify diverse sound textures within tracks, highlighting the significance of capturing texture diversity for enhancing classification performance. A parallel CNN network structure was also devised to extract features from music.^(16,17,18) Features like non-negative matrix factorization and Short-Time Fourier Transform and pitch are extracted and are then subjected to a classification process via Deep Convolutional Neural Network (DCNN) model. To improve the classification accuracy, the DCNN model is trained using a new Self Adaptive model through optimizing the weight. CNN-based music classifier named Bottom-up Broadcast Neural Network (BBNN) is trained on STFT spectrograms extracted from the music clip. An accuracy of 97,51 % and 74,39 % over the GTZAN and the FMA dataset respectively.^(19,20)

Literature surveys reveal that most of the research is carried out in music genre classification mostly using same datasets on machine learning and deep learning models and recently some research was carried out on Western music classification. However, no research has been carried out in classification of semi or light classical music of the North Indian music genre. Even though varieties of music genres exist in North Indian classical music due to cultural diversity, in this research only eleven kinds of semi-classical music are chosen for classification purposes. The accuracy achieved on these models still needs a lot of improvement and there is huge scope to experiment with new models on different genres of music which are untouched by the past researchers. This is novel research using deep learning with dual optimized ANN architecture integrated with Cat Swarm and Optimal Brain Damage for pruning. Proposed Architecture is shown in figure 1.

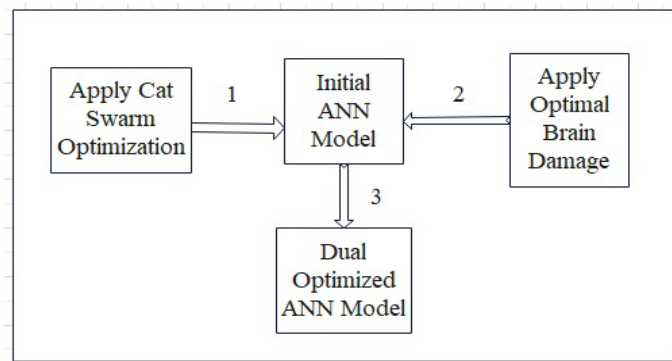


Figure 1. Architecture Diagram

METHOD

ANN Commonly termed as Artificial Neural Network; it is a simulation of the natural nervous system found in humans. ANN exhibits adaptability, learning, generalization, fault tolerance, and self-organizing features which are very interesting to study. It can solve complex problems by applying massive parallel processing. ANN structure consists of hidden layers, input and output layers, and connection weights and links. The processing units of ANN are neurons called MLP (Multi Layer Perceptron). The number of parameters can be varied based on the complexity of the problem. Training with ANN involves trying out different values of weights until the optimum output is obtained. It involves identifying the optimum set of weights to reach a realistic output that closely resembles the actual target. Building optimal ANNs that can output high accuracy is a challenging research task. The efficiency of the ANN can be improved still further by choosing appropriate optimization algorithms.

Optimizing ANN - Using swarm intelligence- Swarm Intelligence (SI) is the influence of swarming traits exhibited by a cluster of living creatures like bees, ants, termites, birds, and fish in acquiring and passing on knowledge among each other. Swarms can optimally utilize their surroundings and resources applying their folk intelligence. Swarm intelligence algorithms apply stochastic search techniques. These algorithms are proven to be effective, adaptable, and resilient, as well as yielding close results and applying implied concurrent techniques. In this research, for music genre classification, a cat population-based optimization algorithm known as Cat Swarm Optimization is used. It is a robust swarm intelligence-oriented optimizing technique introduced by Chu et al. in the year 2006 is used to get very high accuracy in ANN.

Using Neural Network Pruning -ANN can be further optimized with some pruning algorithms to minimize the complexity of its network architecture without deteriorating its classification and prediction capability. A very large network over-fits the training data resulting in excessive processing capacity and a very small network underfits the training data due its insufficient processing capacity and both the conditions produces adverse ANN which is less adaptive to unknown samples. Determining favourable size for ANN which can produce optimum output is usually conundrum. It takes a lot of trial and error to figure it out. Another way to boost the classification performance of ANNs is to train the neural network which is usually larger than it needs to be and then trim the extra components. Network pruning begins with a big ANN architecture, which has a greater number of hidden neurons. While pruning large ANNs, redundant hidden layers, neurons, and connection weights are removed. In general, each stage of the pruning procedure removes one connection weight or neuron until optimal neural network architecture is obtained. The neurons unresponsive to variations in error are eventually deleted in every iteration. resulting in a network that is considerably smaller and has a greater performance capability with reduced complexity and improved speed of convergence. Pruning is based on the assessment of the overall sensitivity of the neurons to the error and its importance of significance called saliency of the connection weight. Pruning retains important weights. The Optimal Brain Damage (OBD) algorithm is used in this research to prune the CSO-optimized ANN to a further level. The implementation details of CSO and OBD algorithms are discussed in the following sections.

Establish the number of layers, neurons in each layer, and activation functions in the first step of the definition of the neural network architecture. Next, set the network's initial weights and biases. Set the CSO parameters in the second phase. To do this, the neural network's parameters, such as weights and biases, are mapped into a format that can be optimised by CSO. Define a fitness function that measures some metric (accuracy or error, for example) to assess the network's performance. The neural network's parameters should then be evolved using the specified Cat Swarm Optimisation. Virtual cats are moved around the search space during the optimization process to find the best values for the network parameters. then reverse-engineer the optimized parameters that were acquired from CSO.

One of the criteria for removing unimportant neurons is to compute the output sensitivity of the network relative to the activation of each neuron. Neurons that have little effect on the output of the network as a whole are seen as less important. This is accomplished by setting a threshold depending on the effect of unimportant

neurons on the network's performance to recognize them. This criterion might consider the amount to which each neuron influences the network or the shift in accuracy. If the neuron drops below the predetermined threshold, it is either eliminated or pruned. Take appropriate action by eliminating individual neurons and their corresponding connections from the network design. To prevent overfitting or excessive pruning, validate the network's performance regularly while going through these procedures. Adjust hyperparameters as necessary, particularly in the trimming stage. Further modifications could be made considering the feature set.

DEVELOPMENT

Proposed Algorithm

Initialize Neural Network

Define the initial architecture of the Artificial Neural Network (ANN) with suitable layers and neurons.
Encode the parameters of the network for Cat Swarm Optimization (CSO).

CSO Optimization

Apply Cat Swarm Optimization to evolve the parameters of the neural network.
Use a fitness function to evaluate the performance of the network.

Train CSO-Optimized Model

Decode the optimized parameters obtained from CSO back into the neural network architecture.
Train the CSO-optimized model using the training dataset.

Optimal Brain Damage (OBD)

Calculate the sensitivity of the loss with respect to each weight in the network.
Prune the least significant weights based on their sensitivity.
Evaluate and retrain the pruned model to maintain or slightly degrade performance.

Eliminate Insignificant Neurons

Calculate the sensitivity of the network's output concerning each neuron's activation.
Establish a threshold to identify insignificant neurons based on their impact on the network.
Remove or prune neurons that fall below the established threshold.

Final Evaluation

Evaluate the final pruned model on a separate test dataset to assess its performance.
Fine-tune hyperparameters as needed and validate against the desired classification goals.
Adjust the parameters and thresholds based on experimental results and the dataset's characteristics.

Table 1. ANN Initialization	
Sl.No.	Parameter Value
1	Input Neurons:10
2	Hidden Layers :1
2	Hidden Neurons: 15
3	Output Neurons: 11
Hyper Parameters	
4	Learning Rate:0,001
5	Activation Function (for hidden layer): ReLU
6	Activation Function (for output layer): softmax
7	Loss Function: categorical cross-entropy
8	Number of Epochs:100
9	Batch Size:32
10	Weights Initialization: Random initialization
11	Biases Initialization: Random initialization
12	Regularization: L2
13	Dropout: 0,5
14	Optimization function: Adam
15	Metrics: Accuracy

Steps involved in Cat Swarm Optimization (CSO)

1. Initialization: initialize a swarm of cats. Each cat represents a candidate solution in the search space. Randomly initialize the position and velocity of each cat. Evaluate the fitness of each cat based on the objective function.
2. Update cat Positions and Velocities: update the velocity of each cat using the current velocity, cognitive component (individual's best-known position), and social component (swarm's best-known position). Update the position of each cat using the updated velocity. Ensure that the new position is within the search space boundaries.
3. Evaluate Fitness: evaluate the fitness of each cat based on the updated position using the objective function.
4. Update Individual and Swarm Best Positions: update the best-known position of each cat if its fitness improves. Update the best-known position of the entire swarm if any cat achieves better fitness.
5. Update Inertia Weight: optionally, update the inertia weight. This helps control the trade-off between exploration and exploitation.
6. Check Stopping Criteria: check if the maximum number of iterations or a convergence criterion is met. If yes, stop the optimization process; otherwise, go back to step 2.
7. Output: the best-known position of the entire swarm represents the optimized solution.

Algorithm

1. Initialize swarm
2. Evaluate fitness of each particle
3. While stopping criterion not met:
For each particle in the swarm:
Update velocity and position.
Evaluate fitness.
Update individual best-known position.
Update swarm's best-known position (global)
Update inertia weight (optional)
Output the best-known position of the swarm.

Table 2. CSO Initialization	
Sl.No.	CSO Parameter Value
1	Swarm Size: 30
2	Maximum Iterations: 100
3	# Lower and upper bounds for parameter lower_bound = -1,0 upper_bound = 1,0
4	Dimensionality: dimensionality = (input_neurons * hidden_neurons) + hidden_neurons + (hidden_neurons * output_neurons) + output_neurons with 10 input neurons, 15 hidden neurons, and 11 output neurons, the dimensionality would be $(10 * 15) + 15 + (15 * 11) + 11 = 352$
5	Inertia Weight (w): 0,5 This is a parameter that controls the trade-off between exploration and exploitation.
6	Cognitive Coefficient (c1): 2 This controls the impact of the individual's best-known position on its movement.
7	Social Coefficient (c2): 2 This controls the impact of the swarm's best-known position on the movement of individuals.
8	Velocity Limit: 5,0 This is the maximum allowed velocity for a particle. You can set an upper limit to prevent particles from moving too fast.
9	Random numbers: r1 = np.random.rand(swarm_size, dimensionality) r2 = np.random.rand(swarm_size, dimensionality)

Dual Optimizing ANN Architecture with CSO and OBD-Using Cat Swarm Optimisation (CSO) and the training dataset optimize the ANN's parameters. Determine the sensitivity of the loss relative to each weight after training. This entails calculating Hessian values or second-order derivatives to determine how each weight affects the total loss. The weights are arranged in ascending order of their sensitivity ratings. Which weights have the least effect on the overall loss will be determined with the aid of this sorting. Choose the top 20 %

of the weights with the lowest sensitivity to establish a pruning threshold. Which weights are deemed least significant are determined by this criterion. Decide which weights should be pruned in light of the established threshold. These are the weights from the network that will be removed. Set the indicated weights to zero or eliminate any links to prune them. Through the removal of less important links, this stage lowers the complexity of the model. To make sure that performance is maintained or just marginally decreased, retrain the trimmed model. Analyze the accuracy and generalization capacity of the pruned model using a validation dataset. For even more refinement, one can choose to prune and retrain several times. The trade-off between minimizing performance loss and reducing model size can be improved by this iterative method. By using these procedures, the neural network is pruned in Optimal Brain Damage to increase its efficiency while maintaining its classification abilities. Hyperparameters and thresholds should be adjusted according to the particulars of your model and dataset.

Algorithm for Using OBD (Optimal Brain Damage) to Trim the ANN

1. Compute Loss Sensitivity: determine the sensitivity of the loss with respect to each weight once the CSO-optimized network has been trained.
2. Trim Weights: sort the weights according to sensitivity and eliminate the ones that don't matter as much. Pruning entails eliminating connections entirely or setting specific weights to zero.
3. Assess the pruned network: to guarantee that performance is preserved or only marginally decreased, retrain the trimmed network.
 1. Removing Superfluous Neurones and Standards:
4. Sensitivity of Neuron Activation: determine how sensitive the network's output is to the activation of each neuron. Neurons that have little effect on the output of the network as a whole are seen as less important.
5. Requirements for Removal: determine a cutoff point or set of standards for classifying inconsequential neurons according to how they affect the network's overall performance.
6. Removal of Neurons: neurones that are below the set threshold should be removed or pruned. Take appropriate action by eliminating individual neurons and their corresponding connections from the network design. Taking into account to prevent overfitting or excessive pruning, validate the network's performance regularly while going through these procedures. Adjust hyperparameters as necessary, particularly in the trimming stage. Based on the findings of the experiment, adjustments might be required.

Steps

1. Utilise training data to train a CSO-optimized ANN.
2. Determine the sensitivity of loss for each weight.
3. Weights are sorted in ascending order by sensitivity.
4. Establish the pruning threshold (top 20 % of the weights with the lowest sensitivity, for example).
5. Determine which weights, in light of the threshold, to prune.
6. Eliminate links or set values to zero to prune weights that have been found.
7. Retrain the trimmed model to preserve performance or marginally deteriorate it.
8. Assess the refined model by employing a validation dataset.
9. For more refinement, prune and retrain iteratively.

Calculation of sensitivity

In the context of neural network pruning, sensitivity refers to a neuron's significance or effect on the performance of the network as a whole. A frequently employed measure for computing sensitivity is predicated on the gradients of the loss function concerning the output of the neuron.

Steps to Check the Sensitivity of Each Neuron

1. Train the neural network model on the training dataset using the loss function MSE (Mean Squared Error) and CSO optimization algorithm until convergence.
2. After training, calculate the sensitivity of the loss with respect to the activation of each neuron using backpropagation. Compute the partial derivative of the loss function with respect to the activation of each neuron.
3. Utilize backpropagation to compute the gradients of the loss with respect to the activations of each neuron. This involves applying the chain rule of calculus to propagate sensitivities backward through the network.
4. Average or aggregate the sensitivity values across all samples in the training dataset. This step ensures that the sensitivity values are representative of the entire dataset and not biased by individual samples.

5. Rank the neurons based on their sensitivity values in descending order. Neurons with higher sensitivity values are considered more influential.
6. Determine a sensitivity threshold based on the criteria. This threshold will be used to decide which neurons are significant enough to be retained.
7. Based on the sensitivity rankings and the threshold, decide which neurons to retain and which to discard. Neurons with sensitivity values below the threshold may be candidates for pruning.
8. If pruning is performed, retrain the model on the pruned architecture. This step ensures that the remaining neurons adapt to the changed architecture and that the model maintains its performance.
9. Evaluate the pruned model on a separate test dataset to ensure that the pruning process did not significantly degrade performance.
10. Optionally, iterate through the sensitivity analysis and pruning process multiple times to refine the model further.

$$\text{Hidden Layer Activation} = \text{Input} * \text{Weights} + \text{Bias} \quad (1)$$

$$\text{Hidden Layer Output} = \text{Apply Activation Function}(\text{Hidden Layer Activation}) \quad (2)$$

$$\text{Output} = \text{Hidden Layer Output} * \text{Output Weights} + \text{Output Bias} \quad (3)$$

Sensitivity of neuron with respect to CSO is given as:

$$\text{Sensitivity} = \frac{\text{Fitness}(\text{Personal Best}) - \text{Fitness}(\text{Current Position})}{\text{Fitness}(\text{Personal Best})} \quad (4)$$

Sensitivity represents the improvement achieved by each neuron from its personal best to its current position relative to its personal best fitness. A higher sensitivity indicates a more significant improvement.

In the context of neural network pruning, sensitivity refers to a neuron's significance or effect on the performance of the network. A frequently utilised technique for evaluating sensitivity rests on the gradients of the loss function associated with the output of the neuron. The sensitivity of each neuron is calculated using formula (5).

$$S_i = \frac{\delta \text{Loss}}{\delta \text{Output}_i} \times \text{Output}_i \times (1 - \text{Output}_i) \quad (5)$$

S_i is the sensitivity of the i^{th} neuron.

Where:

$\delta \text{Loss} / (\delta \text{Output}_i)$ is the partial derivative of the loss function with respect to the output of the i^{th} neuron.

Output i is the Output of the i^{th} neuron.

$1 - \text{Output}_i$ is the complement of the output of the i^{th} neuron.

The threshold value of sensitivity is set to 0,03.

Table 3 shows the dataset size of each music genre considered for training and testing.

Table 3. Dataset Size		
Music Genre	Training Size	Testing Size
Chaiti	235	75
Natya Sangeet	473	112
Bhajan	673	187
Qawwali	479	117
Keertan	654	175
Kajri	552	134
Thumri	642	163
Tappa	557	137
Dadra	445	105
Dhun	672	168
Ghazal	538	123

RESULTS

The performance of the effectiveness of the proposed methodology of applying CSO with OBD pruning on ANN was measured in two scenarios. i. Before applying CSO and OBD ii. After applying CSO and OBD.

Before CSO and OBD pruning - The experiment is carried out in the beginning with ANN using the initial parameters. Figure 2 shows the classification report of the music genre classification before applying CSO and OBD. The lowest precision score of 89 % is obtained on Chaiti music genre and 98 % precision is obtained on Bhajan genre. Dadra and Ghazal genre showed 94 % recall rate and all the rest of the genre showed 95 % recall rate. The lowest F1 score of 92 % is achieved by Chaiti genre and highest of 96 % F1 score on Bhajana. An overall accuracy of 95 % is achieved. Figure 3 shows confusion matrix before applying CSO and OBD with. From the total elapsed timing for each epoch as shown in figure 4 , it is observed that maximum time of 6,345 seconds is consumed by epoch1, and minimum of 3,132 seconds are consumed by epoch 10. Mean Square Error (MSE) for each genre before applying CSO and OBD is shown in figure 5. The lowest of MSE of 0,159.

Classification Report:				
	precision	recall	f1-score	support
Chaiti	0.89	0.95	0.92	75
Natya Sangeet	0.94	0.95	0.94	112
Bhajan	0.98	0.95	0.96	187
Qawwali	0.95	0.95	0.95	117
Keertan	0.97	0.95	0.96	175
Kajri	0.95	0.95	0.95	134
Thumri	0.96	0.94	0.95	163
Tappa	0.94	0.95	0.95	137
Dadra	0.92	0.94	0.93	105
Dhun	0.95	0.95	0.95	168
Ghazal	0.93	0.94	0.94	123
accuracy			0.95	1496
macro avg	0.94	0.95	0.94	1496
weighted avg	0.95	0.95	0.95	1496

Figure 2. Classification report before applying CSO and OBD

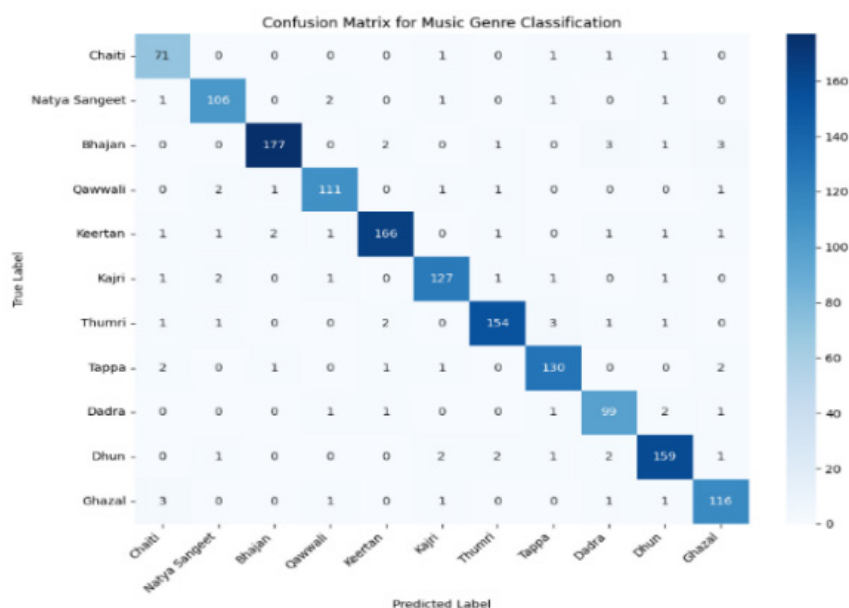


Figure 3. Confusion matrix before applying CSO and OBD

Epoch 1: Total Elapsed Time = 6 seconds 345 milliseconds
 Epoch 2: Total Elapsed Time = 5 seconds 988 milliseconds
 Epoch 3: Total Elapsed Time = 5 seconds 631 milliseconds
 Epoch 4: Total Elapsed Time = 5 seconds 274 milliseconds
 Epoch 5: Total Elapsed Time = 4 seconds 917 milliseconds
 Epoch 6: Total Elapsed Time = 4 seconds 560 milliseconds
 Epoch 7: Total Elapsed Time = 4 seconds 203 milliseconds
 Epoch 8: Total Elapsed Time = 3 seconds 846 milliseconds
 Epoch 9: Total Elapsed Time = 3 seconds 489 milliseconds
 Epoch 10: Total Elapsed Time = 3 seconds 132 milliseconds

Figure 4. Total elapsed time for each epoch before CSO and OBD

Mean Squared Error (MSE) for each genre before applying CSO and OBD
 Chaiti: 0.49422232030099805
 Natya Sangeet: 0.2108468389906792
 Bhajan: 0.2100137187827888
 Qawwali: 0.15955590914175155
 Keertan: 0.5330230283542614
 Kajri: 0.4583488086862573
 Thumri: 0.36211681070233964
 Tappa: 0.4871472458019339
 Dadra: 0.45219656954638987
 Dhun: 0.287678796695538
 Ghazal: 0.4690483744682411

Figure 5. MSE for each music genre before CSO and OBD

After CSO OBD - The ANN model is fitted with CSO, and model is run. The sensitivity of each neuron is noted and any neuron with a sensitivity value less than 0,03 is pruned from the ANN. Table 4 shows the details of sensitivity value and pruning decision applies on each neuron.

Neuron No.	Initial Position	Initial Weights (w1, w2, w3)	Initial Bias	Sensitivity	Pruned (Yes/No)
1	[0,1, 0,2, 0,3]	[0,4, -0,1, 0,8]	0,03	-0,024	Yes
2	[-0,3, 0,5, 0,2]	[-0,2, 0,7, -0,1]	-0,05	0,042	No
3	[0,6, -0,1, 0,7]	[0,9, 0,6, 0,4]	0,07	0,048	No
4	[0,2, -0,1, 0,5]	[0,6, 0,4, -0,3]	-0,01	0,008	Yes
5	[0,3, -0,4, 0,6]	[0,1, 0,5, -0,2]	0,05	-0,012	Yes
6	[0,4, 0,3, -0,1]	[0,2, -0,2, 0,5]	-0,03	0,035	No
7	[-0,2, 0,6, 0,1]	[0,2, -0,3, 0,7]	0,01	0,015	Yes
8	[0,7, 0,1, -0,3]	[-0,2, -0,5, 0,4]	-0,04	0,043	No
9	[0,6, -0,3, 0,1]	[0,5, 0,2, -0,7]	0,05	-0,012	Yes
10	[-0,3, 0,4, -0,2]	[0,1, 0,2, -0,4]	-0,04	0,021	Yes
11	[0,4, 0,1, -0,6]	[-0,3, 0,7, 0,2]	-0,05	0,032	No
12	[-0,1, -0,3, 0,5]	[0,6, 0,2, -0,4]	0,02	-0,008	Yes
13	[0,3, 0,5, -0,2]	[0,2, -0,4, 0,6]	-0,04	0,032	No
14	[0,1, 0,2, -0,3]	[-0,4, 0,6, -0,2]	0,02	-0,028	Yes
15	[0,7, 0,2, -0,4]	[0,3, -0,1, 0,2]	0,04	-0,012	Yes

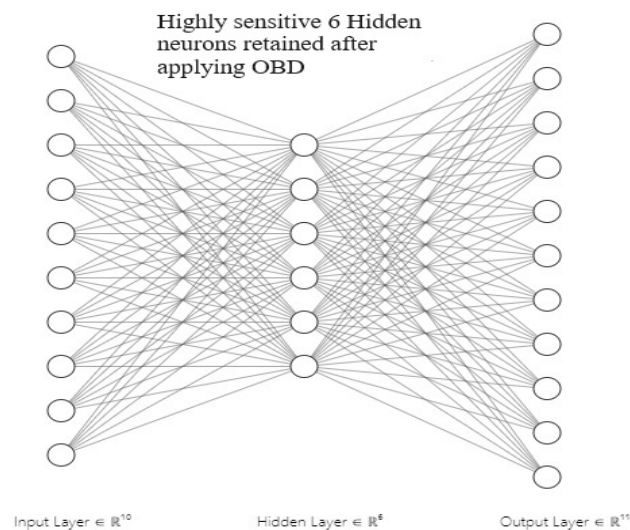


Figure 6. OBD Optimized pruned ANN

Classification Report:				
	precision	recall	f1-score	support
Chaiti	0.94	0.97	0.95	75
Natya Sangeet	0.96	0.98	0.97	112
Bhajan	0.99	0.98	0.99	187
Qawwali	0.97	0.98	0.98	117
Keertan	0.98	0.98	0.98	175
Kajri	0.98	0.98	0.98	134
Thumri	0.99	0.98	0.99	163
Tappa	0.99	0.98	0.98	137
Dadra	0.97	0.98	0.98	105
Dhun	0.99	0.98	0.99	168
Ghazal	0.98	0.98	0.98	123
accuracy			0.98	1496
macro avg	0.98	0.98	0.98	1496
weighted avg	0.98	0.98	0.98	1496

Figure 7. Classification report after applying CSO and OBD

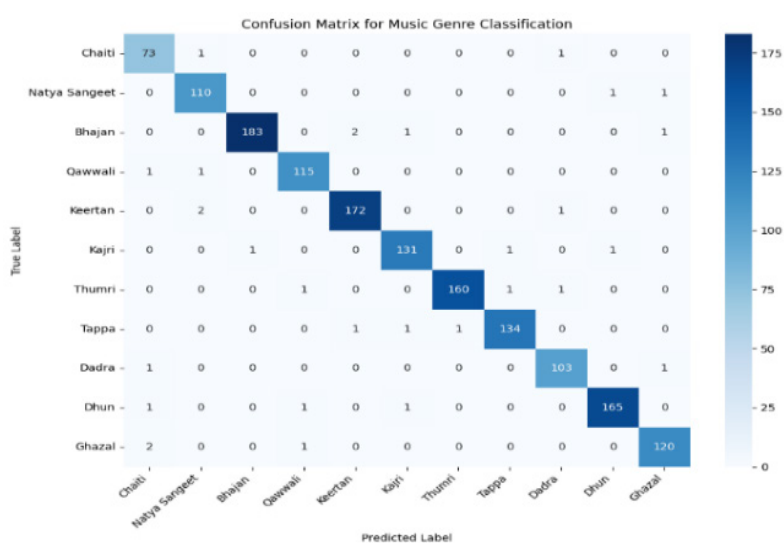


Figure 8. Confusion matrix after applying CSO and OBD

Epoch 1: Total Elapsed Time = 3 seconds 85 milliseconds
 Epoch 2: Total Elapsed Time = 2 seconds 473 milliseconds
 Epoch 3: Total Elapsed Time = 1 seconds 666 milliseconds
 Epoch 4: Total Elapsed Time = 1 seconds 868 milliseconds
 Epoch 5: Total Elapsed Time = 1 seconds 820 milliseconds
 Epoch 6: Total Elapsed Time = 1 seconds 894 milliseconds
 Epoch 7: Total Elapsed Time = 2 seconds 26 milliseconds
 Epoch 8: Total Elapsed Time = 3 seconds 157 milliseconds
 Epoch 9: Total Elapsed Time = 2 seconds 102 milliseconds
 Epoch 10: Total Elapsed Time = 2 seconds 843 milliseconds

Figure 9. Elapsed time for each epoch after CSO and OBD

Mean Squared Error (MSE) for each genre after applying CSO and OBD
 Chaiti: 0.16769997984046398
 Natya Sangeet: 0.04365948458728251
 Bhajan: 0.05720270029632647
 Qawwali: 0.19262536465362168
 Keertan: 0.17651061209766164
 Kajri: 0.22410183428842814
 Thumri: 0.12168056854160424
 Tappa: 0.03952497170068532
 Dadra: 0.20254571771312566
 Dhun: 0.1280730800669747
 Ghazal: 0.14136217762400205

Figure 10. MSE of each genre after CSO and OBD

The final dual-optimized architecture of the proposed model consists of 10 input neurons, 6 hidden neurons with highly sensitive value which can effectively contribute to the training process. Figure 6 shows highly optimized pruned ANN after applying OBD. Figure 7 shows the classification report where it is observed the precision, recall and f1-score of each genre improved to the highest rate of 99 % for few genres as shown in figure 6. Also, improved average accuracy of 98 % is obtained. Figure 8 shows the confusion matrix with an increase in True Positive rate for every music genre. Figure 9 shows the total elapsed time of each epoch with an approximate fifty percent reduction of processing time in each epoch with the lowest of 1,666 seconds. Figure 10 shows MSE values of each genre after CSO and OBD. The lowest MSE of 0,0395 is obtained. The MSE of each genre is reduced in comparison with the MSE before applying CSO and OBD.

DISCUSSION

Comparative Analysis of Results - figure 11 shows the graph of comparison of elapsed time of each epoch before and after CSO and OBD. Figure 12 shows the bar chart of MSE of each genre before and after CSO and OBD. MSE of each genre decreased after applying CSO and OBD. This is clearly seen in the graph. Figure 13 shows the accuracy graph of ANN before and after applying CSO and OBD. It is observed that accuracy of ANN after CSO and OBD increased to 98 %. The comparison of validation loss before and after applying CSO and OBD is shown in figure 14. The loss decreased to the lowest of 0,02 after applying CSO and OBD. The proposed model is also tested with GTZAN dataset, Indian Music Genre (IMG) dataset, Hindustan Music Rhythm (HMR) and Tabala Dataset. The model gave accuracy of 98,34 %, 98,23 %, 98,65 % and 98,41 % respectively. This proves the efficiency of the developed model being suitable to apply to any music genre classification.

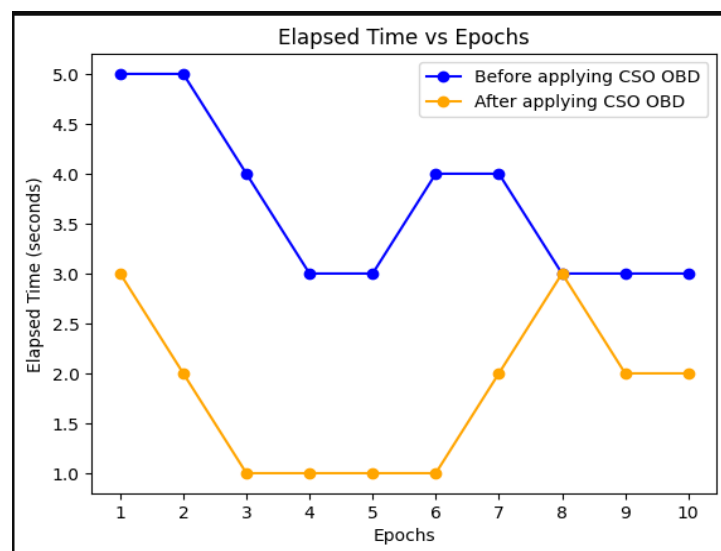


Figure 11. Graph showing elapsed time before and after applying CSO and OBD

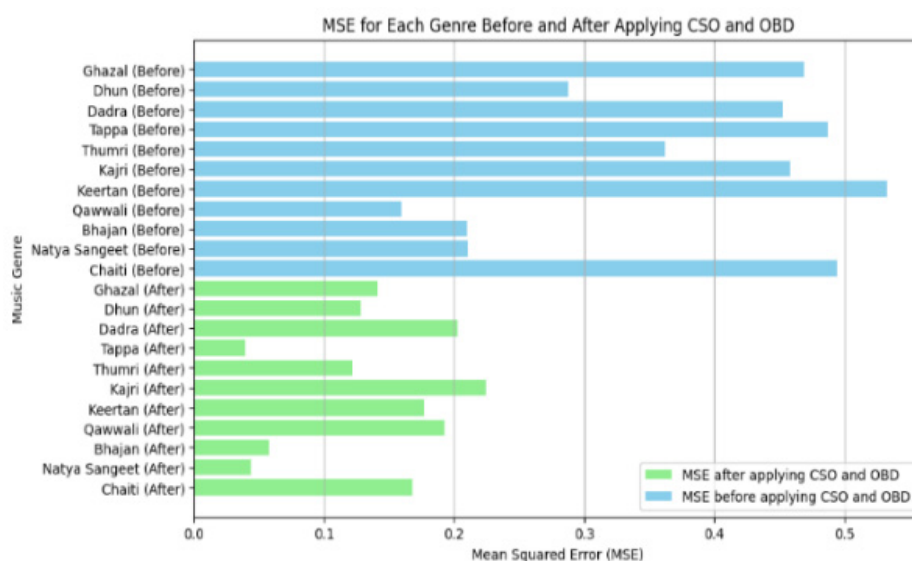


Figure 12. Chart showing comparison of MSE of each genre before and after applying CSO and OBD

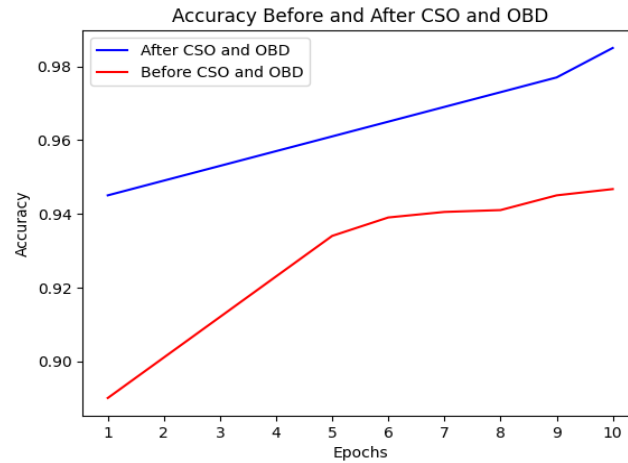


Figure 13. Graph showing accuracy before and after CSO and OBD

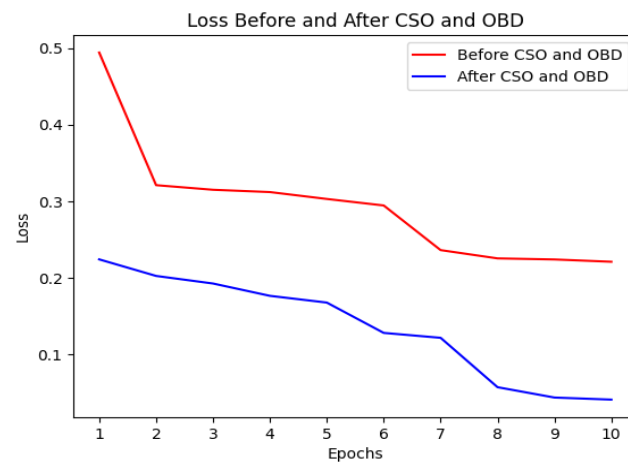


Figure 14. Graph showing loss before and after CSO and OBD

The Advanced Dual-Optimized Neural Network Model with Integrated CSO (Cat Swarm Optimisation) and OBD (Optimal Brain Damage) for Precise Classification and Prediction of North Indian Light Classical Music Genre has improved accuracy and efficiency. The classification accuracy of 98 %, which is a considerable improvement over the prior performance of 95 % without CSO and OBD, demonstrates the effectiveness of the dual optimisation technique. The network architecture has been greatly improved by the combination of CSO and OBD. A more robust optimisation process has been made possible by CSO's capacity to replicate the hunting behaviour of cats. This has helped to obtain the global maximum and fine-tune the model parameters—both of which are essential for getting the best classification accuracy. After OBD implementation, the number of neurons decreased from 11 to 6, indicating a more simplified model that prioritises keeping only high-sensitivity neurons that are essential for precise categorization. This reduction enhances performance speed and predicted accuracy by highlighting the model's capacity to rank pertinent information in a way that maximises computing efficiency.

Sl. No.	Model Used	Accuracy
1	Deep Neural Network ⁽¹⁾	93,50 %
2	Biological Neural Network ⁽²⁾	92,10 %
3	Deep Learning ⁽⁴⁾	91,80 %
4	Convolutional Neural Network ⁽⁹⁾	90,20 %
5	Recurrent Convolutional Neural Network ⁽¹⁸⁾	89,70 %
6	Particle Swarm Optimization	87,50 %
7	Bat Swarm Optimization	88,30 %
8	Ant Swarm Optimization	86,90 %
9	DONN with CSO and OBD (current work)	98,00 %

The suggested framework's superiority is confirmed by comparison with earlier models, demonstrating its effectiveness in correctly classifying musical genres as shown in table 5.

CONCLUSIONS

The combination of OBD and CSO methods emphasises how flexible the model is to the nuances of North Indian Light Classical Music categorization. The observed gains show potential for wider applications in music genre classification in addition to validating the usefulness of the suggested approach. Subsequent investigations could examine how well the model fits other musical cultures, which could provide light on cross-cultural music analysis. The tested dual optimization approach developed in this research can be utilized for any other application to improve the performance of ANN model. To sum up, the results show that CSO and OBD collaborated well in order to attain precise categorization and prediction within the North Indian Light Classical Music genre. Its application of dual optimisation strategies, which prioritises high-sensitivity neurons and makes use of CSO's assistance in reaching the global maximum, marks a major advancement in computational musicology and paves the way for the development of more accurate and efficient algorithms for tasks involving genre classification.

BIBLIOGRAPHIC REFERENCES

1. Zhang K. Music Style Classification Algorithm Based on Music Feature Extraction and Deep Neural Network. *Wireless Communications and Mobile Computing*. 2021;2021:9298654. doi: 10.1155/2021/9298654.
2. Mi D, Qin L. Classification System of National Music Rhythm Spectrogram Based on Biological Neural Network. *Computational Intelligence and Neuroscience*. 2022;2022:2047576. doi: 10.1155/2022/2047576.
3. Patil SA, Pradeepini G, Komati TR. Novel mathematical model for the classification of music and rhythmic genre using deep neural network. *J Big Data*. 2023;10:108. doi: 10.1186/s40537-023-00789-2.
4. Mounika KS, Deyaradevi S, Swetha K, Vanitha V. Music Genre Classification Using Deep Learning. 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA). 2021; Coimbatore, India:1-7. doi: 10.1109/ICAECA52838.2021.9675685.
5. Zheng Z. The Classification of Music and Art Genres under the Visual Threshold of Deep Learning. *Comput Intell Neurosci*. 2022;2022:4439738. doi: 10.1155/2022/4439738.
6. Elbir A, Aydin N. Music genre classification and music recommendation by using deep learning. *Electron Lett*. 2020;56:627-629. doi: 10.1049/el.2019.4202.
7. Zhang W, Zakarya M. Music Genre Classification Based on Deep Learning. *Mob Inf Syst*. 2022;2022:2376888. doi: 10.1155/2022/2376888.
8. Li T. Optimizing the configuration of deep learning models for music genre classification. *Heliyon*. 2024;10(2):e24892. doi: 10.1016/j.heliyon.2024.e24892.
9. Nam J, Choi K, Lee J, Chou S-Y, Yang Y-H. Deep Learning for Audio-Based Music Classification and Tagging: Teaching Computers to Distinguish Rock from Bach. *IEEE Signal Processing Magazine*. 2019;36(1):41-51. doi: 10.1109/MSP.2018.2874383.
10. Athulya KM, Sindhu S. Deep Learning Based Music Genre Classification Using Spectrogram. In: *Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021*. 2021 Jul 10. Available at SSRN: <https://ssrn.com/abstract=3883911>. doi: 10.2139/ssrn.3883911.
11. Qi Z, Rahouti M, Jasim MA, Siasi N. Music Genre Classification and Feature Comparison using ML. In: *Proceedings of the 2022 7th International Conference on Machine Learning Technologies (ICMLT '22)*. 2022. Association for Computing Machinery, New York, NY, USA:42-50. doi: 10.1145/3529399.3529407.
12. Bahuleyan H. Music genre classification using machine learning techniques. *arXiv preprint arXiv:1804.01149*. 2018. Available at: <https://arxiv.org/abs/1804.01149>.
13. Liu C, Feng L, Liu G, Wang H, Liu S. Bottom-up broadcast neural network for music genre classification. *Multimedia Tools Appl*. 2021;80(5):7313-7331. doi: 10.1007/s11042-020-09643-6.

14. Ghildiyal A, Singh K, Sharma S. Music Genre Classification using Machine Learning. 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA). 2020; Coimbatore, India:1368-1372. doi: 10.1109/ICECA49313.2020.9297444.
15. Yu-Huei C, Pang-Ching C, Duc-Man N, Che-Nan K. Automatic music genre classification based on CRNN. *Eng Lett*. 2021;29(1):36.
16. Foleis JH, Tavares TF. Texture selection for automatic music genre classification. *Appl Soft Comput*. 2020;89:106127. doi: 10.1016/j.asoc.2020.106127.
17. Yang R, Feng L, Wang H, Yao J, Luo S. Parallel Recurrent Convolutional Neural Networks-Based Music Genre Classification Method for Mobile Devices. *IEEE Access*. 2020;8:19629-19637. doi: 10.1109/ACCESS.2020.2968170.
18. Kumaraswamy B, Poonacha PG. Deep Convolutional Neural Network for musical genre classification via new Self Adaptive Sea Lion Optimization. *Appl Soft Comput*. 2021;108:107446. doi: 10.1016/j.asoc.2021.107446.
19. El Achkar C, Couturier R, At  chian T, Makhoul A. Combining Reduction and Dense Blocks for Music Genre Classification. In: Mantoro T, Lee M, Ayu MA, Wong KW, Hidayanto AN, editors. *Neural Information Processing. ICONIP 2021. Communications in Computer and Information Science*, vol 1517. Springer, Cham. 2021. doi: 10.1007/978-3-030-92310-5_87.
20. Athulya KM, Sindhu S. Deep Learning Based Music Genre Classification Using Spectrogram. In: *Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021*. 2021 Jul 10. Available at SSRN: <https://ssrn.com/abstract=3883911>. doi: 10.2139/ssrn.3883911.

FINANCING

The authors did not receive financing for the development of this research.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

AUTHORSHIP CONTRIBUTION

Conceptualization: Jyothi N M.

Data curation: Pavani G, Divvela Surendra, Sangita Chakraborty.

Formal analysis: Jyothi N M, Satishkumar Patnala, Katakam Ranga Narayana.

Research: Jyothi N M, Satishkumar Patnala, Katakam Ranga Narayana.

Methodology: Jyothi N M, Satishkumar Patnala, Katakam Ranga Narayana.

Project management: Sangita Chakraborty, Pavani G, Divvela Surendra.

Software: Open-Source Google co lab.

Supervision: Jyothi N M, Satishkumar Patnala, Sangita Chakraborty.

Validation: Jyothi N M, Katakam Ranga Narayana.

Display: Jyothi N M, Satishkumar Patnala.

Drafting - original draft: Pavani G, Divvela Surendra.

Writing - proofreading and editing: Jyothi N M, Pavani G, Divvela Surendra.