

Hyperparameter optimization of convolutional neural network using particle swarm optimization for emotion recognition

Dian Palupi Rini¹, Tri Kurniasari², Winda Kurniasari³, Novi Yusliani¹

¹Department of Informatics Engineering, Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia

²Department of Information Technology, Padang Selasa Public Health Center, Palembang, Indonesia

³Department of Information Systems, Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia

Article Info

Article history:

Received Feb 3, 2024

Revised Jul 9, 2024

Accepted Jul 26, 2024

Keywords:

Convolutional neural network

Particle swarm optimization

Electroencephalography signal

Emotion

Hybrid

ABSTRACT

Emotion identification has been widely researched based on facial expressions, voice, and body movements. Several studies on emotion recognition have also been performed using electroencephalography (EEG) signals and the results also show that the technique has a high level of accuracy. EEG signals that detected by standart method using exclusive representations of time and frequency domains presented unefficient results. Some researchers using the convolutional neural network (CNN) method performed EEG signal for emotional recognition and obtained the best results in almost all benchmarks. Although CNN has shown fairly high accuracy, there is still a lot of room for improvement. CNN is sensitive to its hyperparameter value because it has considerable effect on the behavior and efficiency of the CNN architecture. So that the use of optimization algorithms is expected to provide an alternative selection of appropriate hyper parameter values on CNN. Particle swarm optimization (PSO) algorithm is a metaheuristic-based optimization algorithm with many advantages. This PSO algorithm was chosen to optimize the hyperparameter values on CNN. Based on the evaluation results in each model, hybrid CNN-PSO showed better results and achieved the best value in 80:20 split data which is 99.30% accuracy.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Dian Palupi Rini

Department of Informatics Engineering, Faculty of Computer Science, Universitas Sriwijaya

St. Palembang-Prabumulih KM.32, Ogan Ilir, Palembang, Indonesia

Email: dprini@unsri.ac.id

1. INTRODUCTION

Research on emotion recognition has also been performed using electroencephalography (EEG) signals and the results show that the technique has a high level of accuracy [1]. Through EEG, information about mental activity and human emotional states can be known and the information produces different values when emotional states change. Therefore, the use of EEG is considered effective for emotional identification or prediction [2].

Traditional methods of EEG signal classification, which use separate representations for time and frequency domains, often yield unsatisfactory results. In this approach, EEG signals are decomposed into time-frequency representations using discrete wavelet transform (DWT). Statistical features are then calculated to represent the distribution of the signals [3], [4]. Research on EEG signal recognition using end-to-end models based on convolutional neural networks (CNNs) has been conducted and obtained good results in almost all classification benchmarks [5], [6]. The EEG signal representation problem in the database for emotion analysis

using physiological (DEAP) dataset shows that the proposed method achieves an accuracy of 77.98% on valence recognition and 72.98% on arousal recognition [6].

Although CNN standards have shown considerable accuracy, there is still a lot of room for improvement [7]. CNN performance cannot be expected by applying the same architecture to different types of tasks, it needs to be adjusted the architecture for specific tasks so as to produce better performance [8]. CNN also requires a large number of samples for the training phase. In addition, CNN has many hyperparameters [9] and a wide variety of architectures that are considered a challenge and it is difficult to determine the best value of hyperparameters manually. CNN is sensitive to its hyperparameter value because it has a considerable effect on the behavior and efficiency of the CNN architecture [10]. To attain hyperparameters with better performance, experts must manually configure a set of hyperparameter options. However, different datasets require different models or hyperparameter combinations, making it complicated [11].

Research on efficient optimization of hyperparameter values in CNNs using particle swarm optimization (PSO) resulted in an increase in image recognition accuracy by 0.7%-5.7% using the Alexnet-CNN standard [9]. Research on the use of PSO on CNN, which is one of the basic methods in deep learning has been conducted by several researchers [7], [12], [13]. The use of PSO in the training process aims to optimize the results of the solution vector on CNN in order to improve recognition accuracy. Experiments show that the accuracy that can be achieved in 4 epochs is 95.08%. These results are better than conventional CNNs and deep belief networks (DBNs). Its execution time is also similar to conventional CNN. This study proposes a CNN and PSO deep learning approach to optimize multiple hyperparameters, resulting in higher results [14].

Therefore, based on previous research, researchers will test the CNN method for EEG signal classification to detect types of emotions. Then optimization will be carried out with the PSO method on the initialization of learning rate, epoch, and batch size to improve the accuracy of signal recognition in this study. Next will be made a comparison with the CNN algorithm itself without the optimization process from the PSO.

2. METHOD

2.1. Hybrid modeling of convolutional neural network–particle swarm optimization

The hybrid CNN-PSO algorithm is built to provide many alternatives in determining the value of CNN hyperparameters using the PSO algorithm, so that it is expected to avoid local optimal solutions. CNNs take their name from mathematical linear operations between matrices called convolutions. The architecture of CNN that shown in Figure 1 has many layers, including a convolutional layer, a non-linear layer, a pooling layer and a fully-connected layer. Convolutional and fully-connected layers have parameters but pooling and non-linear layers have no parameters.

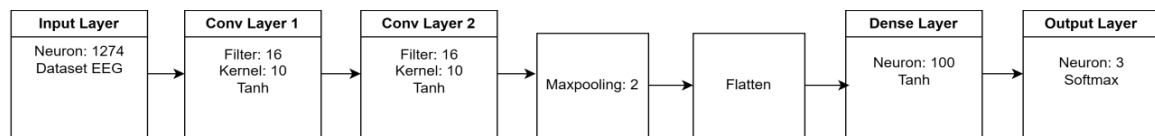


Figure 1. CNN architecture

PSO is a method inspired by the nature of a group of animals such as birds, termites, bees, or ants. Three crucial aspects of PSO include particles, cognitive and social components, and particle velocity. Each particle represents the solution to a given problem. Particle learning includes two elements, which are particle experience (cognitive learning) and the combination of whole swarm learning (social learning) [15], [16]. Cognitive learning in the form of pBest is the best position a particle can ever reach, while social learning in the form of gBest is the best position of all particles in a swarm. The parameters pBest and gBest are used to calculate the speed of particle (1) as well as the speed of calculating the position (2) of the next particle.

Speed update:

$$V_{i,m}^{new} = W \cdot V_{i,m}^{old} + C_1 \times R \times (P_{i,m}^{local\ best} - X_{i,m}^{old}) + C_2 \times R \times (P_m^{global\ best} - X_{i,m}^{old}) \quad (1)$$

Position update:

$$X_{i,m}^{new} = X_{i,m}^{old} + V_{i,m}^{new} \quad (2)$$

Where $V_{i,m}$ is speed of the i th particle at the i th iteration, W is the inertial weight coefficient, C_1, C_2 are the acceleration constants (learning rate), R is a random number (0-1), $X_{i,m}$ is the current position of the i -th particle in the i -th iteration, $Pbest_i$ is the previous best position of the i -th particle, and $Pgbest$ is the best particle among all particles in a group or population. The optimization process is carried out by providing alternative hyperparameter values on CNN a number of particles in the PSO algorithm. It then makes the output value of CNN an objective function of the PSO, by finding the maximum value of several alternative processes in each article [17].

2.2. Dataset

For experimental purposes, the dataset used was EEG data that focused on capturing brainwave patterns associated with various emotions and feelings. This dataset consists of 2,549 variables, of which 2,548 variables contain decimal data, while 1 other variable contains data in the form of strings used as labels. For three minutes under each state (positive, neutral, and negative), data was recorded from two participants, consisting of one male and one female. This data collection process was carried out using an EEG muse headband equipped with dry electrodes to record EEG activity at TP9, AF7, AF8, and TP10 location points [18], [19]. The dataset is shown in Table 1, with the following content specifications:

Table 1. Dataset specifications

Data label	Number of data
Positive	708
Negative	708
Neutral	716
Total	2,132

In the context of 2,549 variables, this study applies an even division to form two datasets, each consisting of 1,275 variables. Of these, 1,274 variables will contain decimal data, while 1 variable will serve as a label. The next step is to train both datasets that have been formed. The first dataset will be named "eeg-emotion-1," while the second dataset will be named "eeg-emotion-2." This process is designed to create two similar but independent subsets of data for training purposes.

2.3. Pre-processing

The data pre-processing that has been done in this study includes data transformation and normalization, preparing it for data analysis and deep learning implementation. In the context of using machine learning algorithms, categorical data cannot be directly processed by those algorithms. Therefore, categorical data needs to be converted into numerical form before it can be used in the analysis process [20], [21]. This research involves using label encoding to convert class variables from string data to numeric values. This transformation process maps three classes of strings, namely neutral to class 0, positive to class 1, and negative to class 2. Next, StandardScaler is applied as a normalization method to the dataset. This process aims to change each feature (column) in the dataset so that it has a mean of zero and a standard deviation of one.

2.4. Convolutional neural network hybrid modeling–particle swarm optimization

The process of modeling data in deep learning will be depends on selecting the right model architecture, building a model structure with appropriate layers, initializing parameters, selecting loss functions and optimizers, as well as training the model using training data by updating parameters through optimization algorithms. After training, the model is evaluated using validation data to check its performance, and hyperparameters can be reset to improve results [19], [22]. Final testing is done using unseen test data. Figure 2 is the complete process of CNN's PSO.

CNN is very effective in understanding visual data, recognizing patterns, as well as extracting features hierarchically [23]–[25]. CNN architecture is a combination of three components; Convolutional layer, which contains a number of filters to apply to the input. Each filter scans the input using point products and delivery methods to generate a number of feature maps in a single convolutional layer [4].

In this study, a one-dimensional CNN architecture was used that has a specific configuration to process data. The architecture consists of a convolution layer with 16 filters, each measuring 10, followed by a maxpooling layer with size 2. The hidden layer of the CNN has 100 neurons and uses the activation function of hyperbolic tangent (tanh). The importance of selecting this activation function is proven by experiments, where the use of the rectified linear unit (ReLU) activation function results in unsatisfactory performance. In this study, PSO was formed with a configuration of 3 particles and 5 dimensions. In this scenario, the PSO

algorithm will use 3 particles as search agents to explore a search space that has 5 dimensions, which includes hyperparameter variables to be optimized.

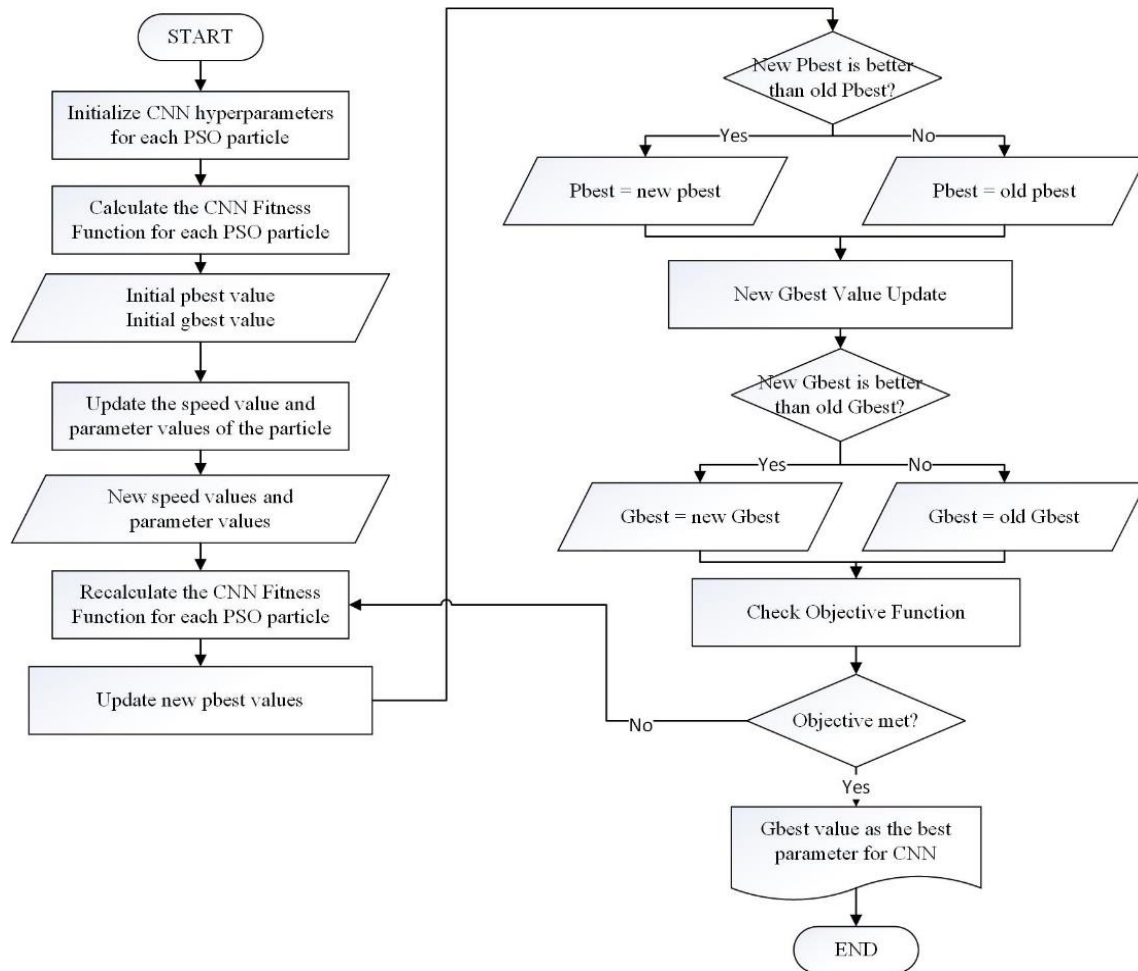


Figure 2. Detail process of CNN-PSO

2.5. Hyperparameter optimizer validation

In this study, the Optunity library was used to perform hyperparameter optimization on a model. Libraries such as optunity and optunity.metrics are imported, as well as the particle swarm solver module from the optunitysolvers library. The search dictionary is used to define the range of hyperparameter values to be explored in the optimization process. Then, the performance function is created with three hyperparameters: lr, epochs, and batchsize.

The validation process of hyperparameter optimization is carried out by considering the range of values on several key parameters, including learning rates ranging from 0.0001 to 0.01, epoch numbers ranging from 1 to 100, and batch sizes ranging from 0 to 1. For batch sizes less than 1, a value of 32 is used, a batch size less than 2 uses a value of 64, and batch sizes not included in both use a value of 128. In addition, there are other parameters such as the number of particles is 3, PSO dimensions are 5, and the number of evaluations is 10.

3. RESULTS AND DISCUSSION

This research involved training ten models, namely five models (models 1-5) on the eeg-emotion-1 dataset and five models (models 6-10) on the eeg-emotion-2 dataset. Each model is trained and evaluated according to the configuration described earlier. Table 2 shows the overall experimental results of the confusion matrix for the eeg-emotion-1 dataset using different data sharing. Meanwhile, Table 3 shows the results of

confusion matrix experiments for the eeg-emotion-2 dataset using different data divisions. It can be seen that 90:10 data division gives the lowest false positive and false negative values.

Table 2. Comparison of confusion matrix eeg-emotion-1 classification results

Ratio data	TP	FP	FN	TN
90:10	212	2	2	424
80:20	415	12	12	830
70:30	621	21	21	1,242
60:40	830	23	23	1,660
50:50	1,033	33	33	2,066

Table 3. Comparison of confusion matrix eeg-emotion-2 classification results

Ratio data	TP	FP	FN	TN
90:10	208	6	6	416
80:20	416	11	11	832
70:30	624	16	16	1,248
60:40	837	16	16	1,674
50:50	1,037	29	29	2,074

3.1. Model 1

The ratio of datasets used in model 1 is 90:10. The training accuracy obtained in model 1 is 99.07. After the data was normalized, the data was trained by adding a PSO algorithm and obtained an accuracy of 98.60. In the case of model 1, CNN coupled with PSO does not affect the accuracy increase. To see the comparison of training and testing per epoch in the CNN accuracy curve, it can be seen in Figure 3, while the loss curve is in Figure 4. Then, the accuracy and loss curves of both training and testing for CNN+PSO can be observed in Figures 5 and 6. Although the accuracy of the CNN model reached 99%, the performance curve showed a less significant improvement, as well as CNN+PSO which showed indications of overfitting on the loss curve, despite obtaining high accuracy [12].

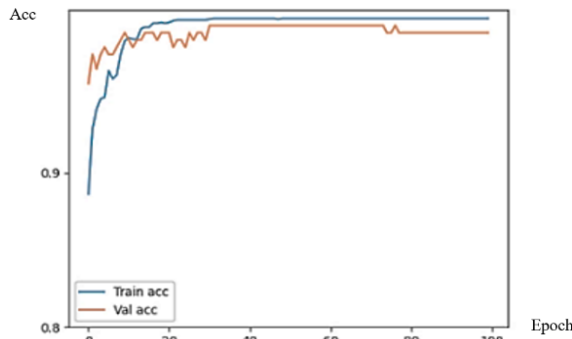


Figure 3. CNN accuracy curve

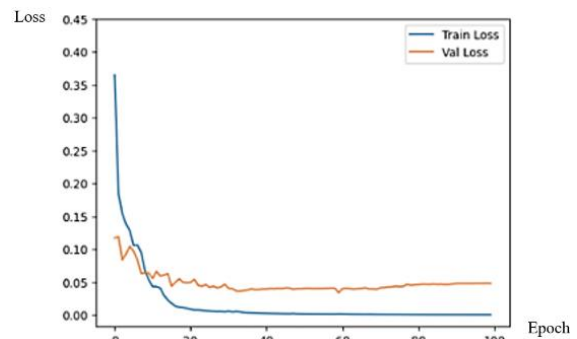


Figure 4. CNN loss curve

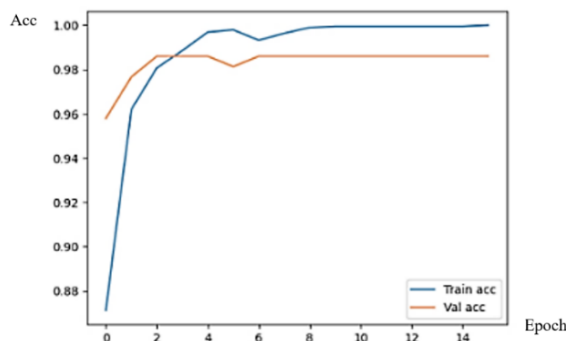


Figure 5. CNN+PSO accuracy curve

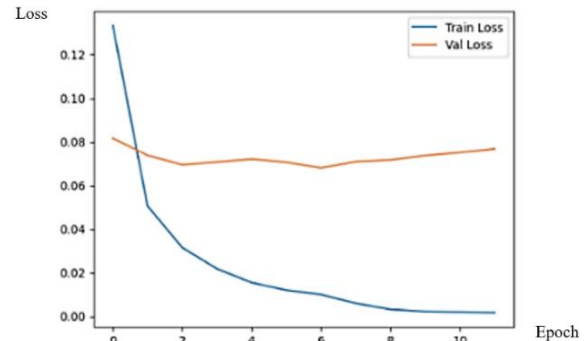


Figure 6. CNN+PSO loss curve

3.2. Model 2

In model 2, training and testing are performed by dividing the data in an 80:20 ratio. The results of the performance evaluation of the CNN in this model showed an accuracy of 97.19%. After going through the data normalization stage, we decided to involve a PSO algorithm to improve performance. The increase resulted in an accuracy of 98.36%. Figures 7 and 8 show the accuracy and loss curves on the CNN, while Figures 9 and 10 represent the accuracy and loss curves on the optimized model using the PSO algorithm. The curve produced by the second model shows less satisfactory quality compared to the first model, with indications of overfitting.

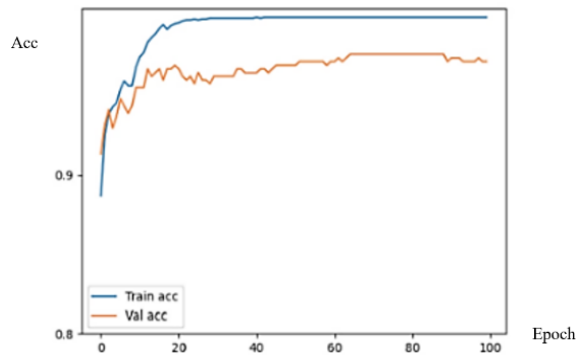


Figure 7. CNN accuracy curve

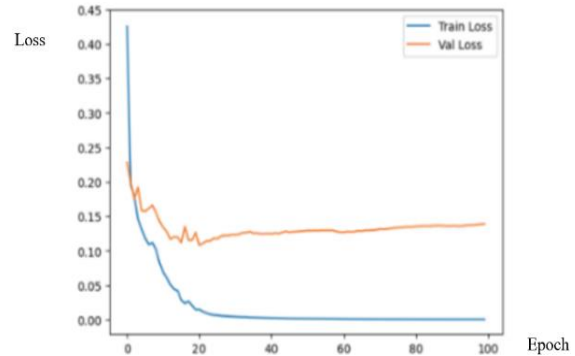


Figure 8. CNN loss curve

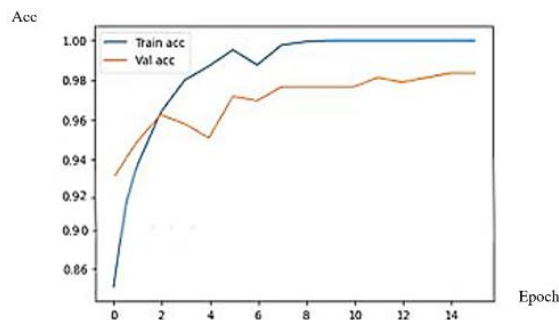


Figure 9. CNN+PSO accuracy curve

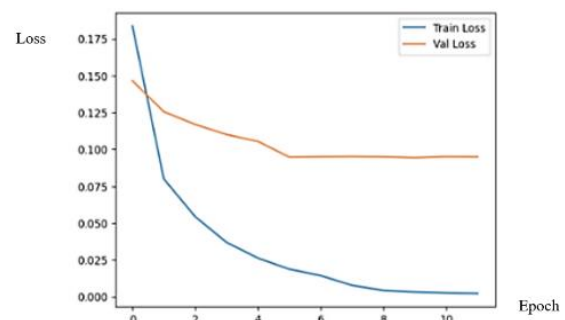


Figure 10. CNN+PSO loss curve

3.3. Model 3

In the third model, we repeated similar steps to the previous model, with a 70:30 variation in data comparison. This model achieved an accuracy of 97.03%, which did not surpass the performance of the two previous models. The accuracy and loss curves can be seen in Figures 11 and 12. After going through the normalization stage, we continued CNN training with the addition of a PSO algorithm for comparison of results. Finally, the accuracy achieved by the CNN+PSO model is 97.19%. In addition, the accuracy and loss curves are also formed which are seen in Figures 13 and 14. The performance of the third model still shows unsatisfactory quality due to overfitting detected, both on the CNN architecture and models that have been improved using the CNN+PSO algorithm.

3.4. Model 4

Furthermore, the fourth model involved training the eeg-emotion-1 dataset with a data comparison of 60:40. Confusion matrix is used to calculate precision, recall, and F1-score with an accuracy of 97.30%. Information on the accuracy and loss curves of the CNN model is presented in Figures 15 and 16. Furthermore, this model gets additional training with the application of PSO algorithms to improve accuracy. This results in an accuracy of 97.66%. The accuracy and loss curves for the CNN+PSO model can be observed in Figures 17 and 18. On the loss chart of the fourth model, train loss shows a positive increase by continuing to decrease near 0. However, this does not apply to loss validation which actually shows an upward trend, even though it remains below the value of 1, resulting in overfitting.

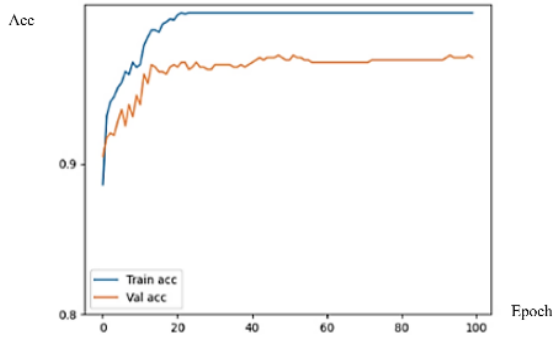


Figure 11. CNN accuracy curve

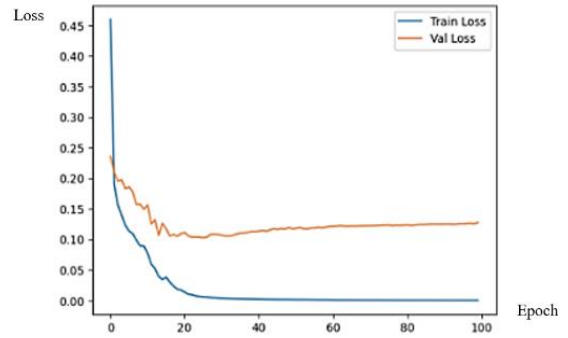


Figure 12. CNN loss curve

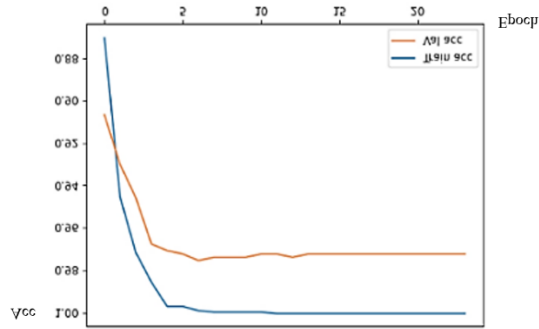


Figure 13. CNN+PSO accuracy curve

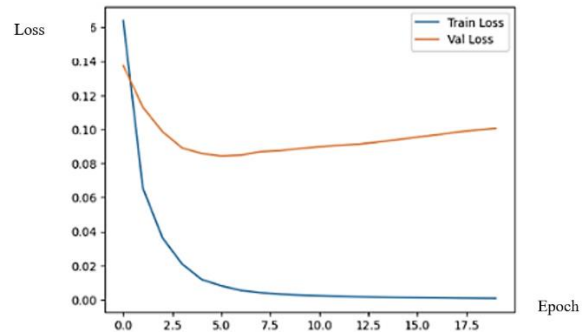


Figure 14. CNN+PSO loss curve

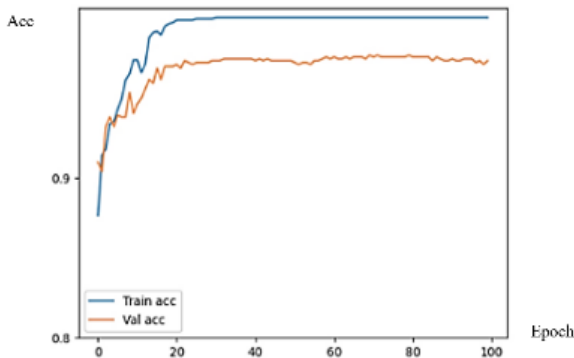


Figure 15. CNN accuracy curve

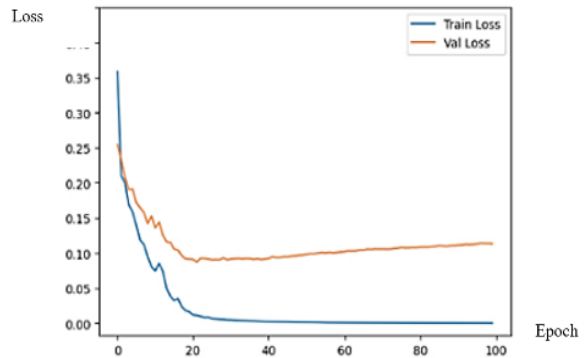


Figure 16. CNN loss curve

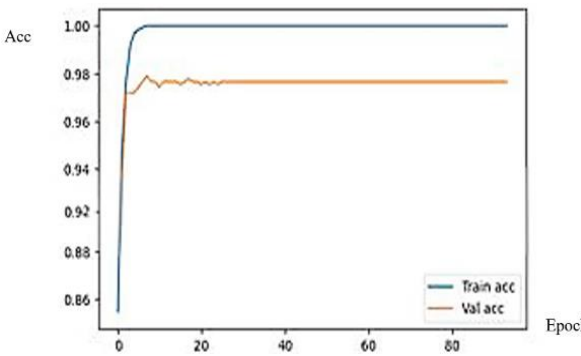


Figure 17. CNN+PSO accuracy curve

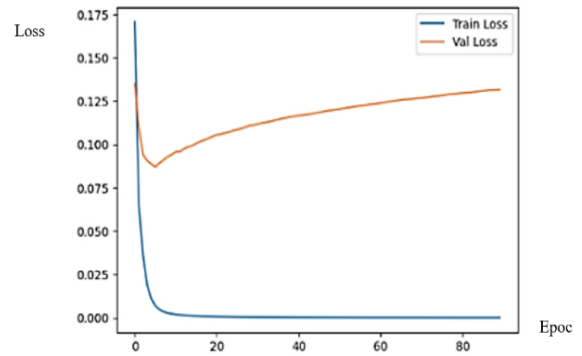


Figure 18. CNN+PSO loss curve

3.5. Model 5

In the fifth model, the data was trained and tested at a ratio of 50:50. By still using the same hyperparameters, the accuracy obtained reaches 96.90%. However, the results from the fifth model showed the lowest accuracy when compared to the four previous models using the eeg-emotion-1 dataset. In addition, an overview of the accuracy and loss curves of the CNN model can be found in Figures 19 and 20. Furthermore, this model gets improved performance through retraining by integrating the PSO algorithm. As a result, accuracy increased by 0.66% to reach 97.56%. Information on the accuracy and loss curves of the CNN+PSO model can be found in Figures 21 and 22. The performance of this model is far behind compared to the other four models, this can be seen from the CNN+PSO loss curve which has a significant difference. Although both train loss and validation loss remain below 0, the results obtained are still unsatisfactory.

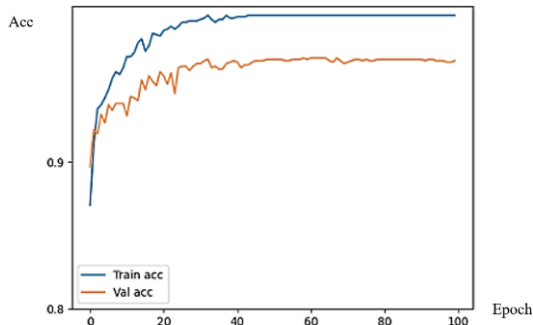


Figure 19. CNN accuracy curve

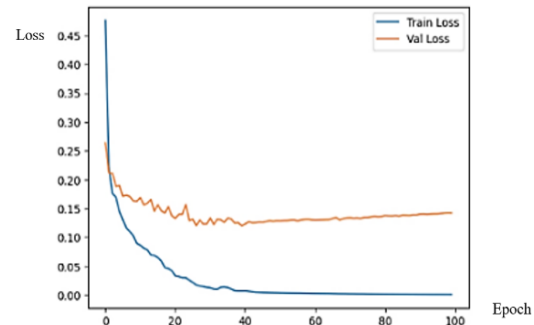


Figure 20. CNN loss curve

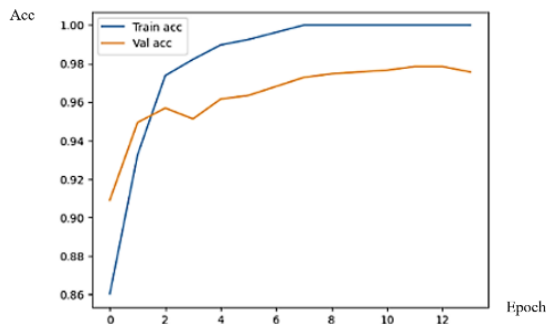


Figure 21. CNN+PSO accuracy curve

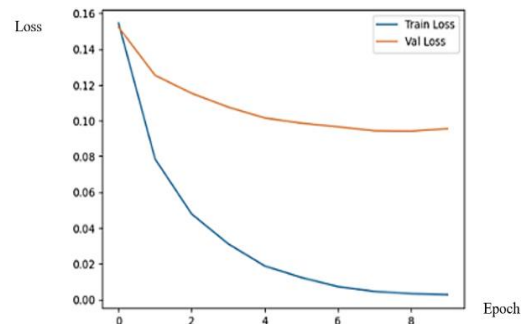


Figure 22. CNN+PSO loss curve

3.6. Model 6

In the sixth model, we trained the eeg-emotion-2 dataset by dividing the data in a 90:10 ratio. By adopting the same parameters as the previous model, the accuracy of CNN on this model reaches 97.20%. The next step involves training the data with the application of the PSO algorithm, which results in an accuracy of 97.66%. This suggests that the use of PSOs can slightly improve accuracy, although not significantly. Comparison of accuracy and loss curves in the CNN model is shown in Figures 23 and 24. Meanwhile, the accuracy and loss curves in the CNN+PSO model are seen in Figures 25 and 26. When compared with the training results of the eeg-emotion-1 dataset using the same data comparison and using both CNN and CNN+PSO architectures, the sixth model resulted in lower accuracy and variable loss values, although still below the value of 0.

3.7. Model 7

In the seventh model, data training was carried out by dividing the dataset in an 80:20 ratio. As a result, the accuracy achieved by this model reaches 97.42%. By involving the PSO algorithm in the next stage of training, accuracy was successfully increased to 99.30%. Comparison with the second model, which involves sharing the same data and identical methods, shows improved performance on the seventh model. However, the resulting performance curve is still not optimal. Details on the CNN accuracy and loss curves are shown in Figures 27 and 28, while the CNN+PSO accuracy and loss curves are shown in Figures 29 and 30.

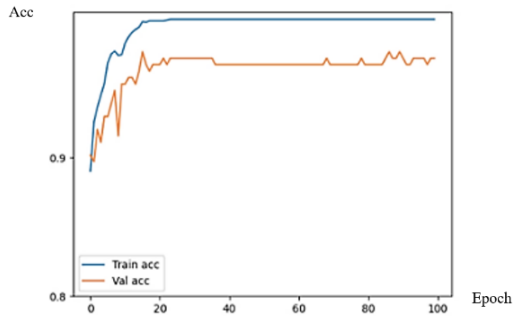


Figure 23. CNN accuracy curve

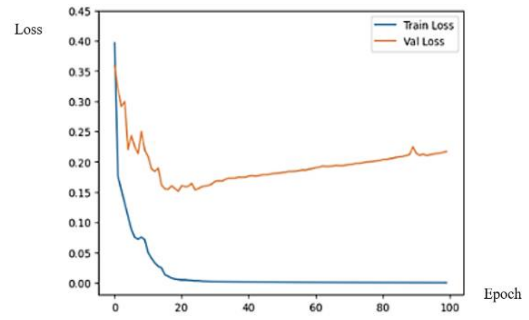


Figure 24. CNN loss curve

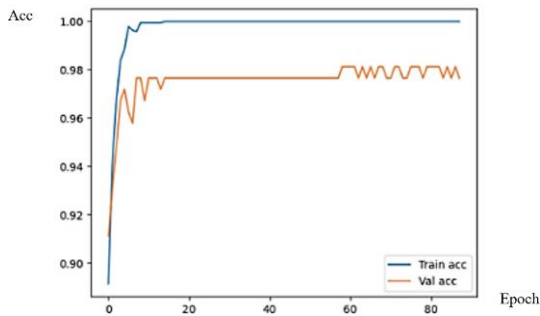


Figure 25. CNN+PSO accuracy curve

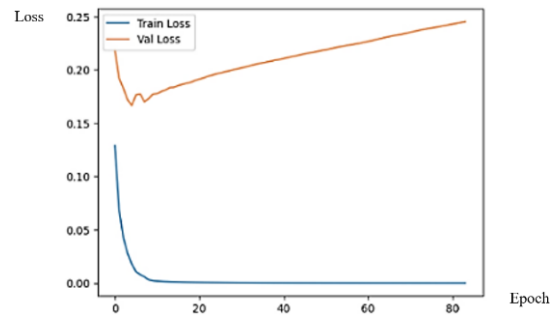


Figure 26. CNN+PSO loss curve

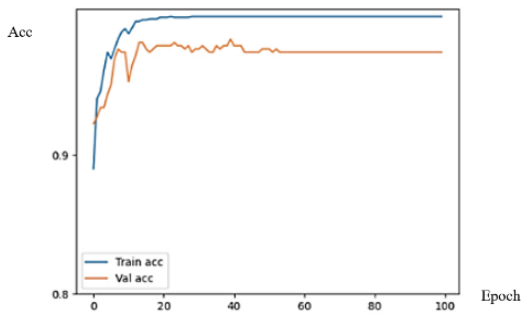


Figure 27. CNN accuracy curve

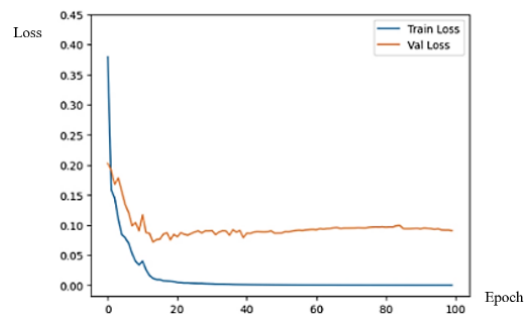


Figure 28. CNN loss curve

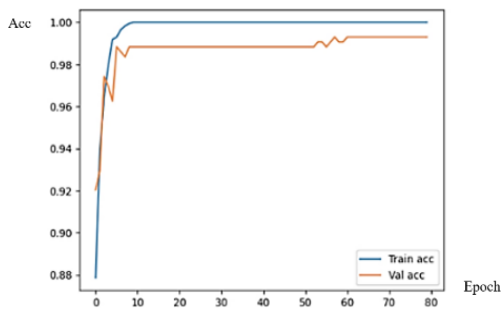


Figure 29. CNN+PSO accuracy curve

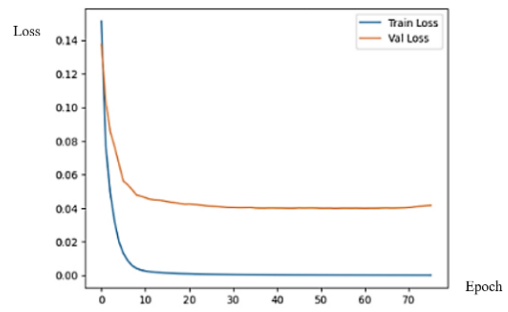


Figure 30. CNN+PSO loss curve

3.8. Model 8

Performance results from the confusion matrix that will be used to calculate precision, recall, and F1-score with an accuracy of up to 97.50%. After the implementation of the PSO algorithm, the accuracy was

successfully increased to 98.12%. Information on the comparison of training and testing per epoch in the accuracy curve of the CNN model is presented in Figure 31, while the loss curve is shown in Figure 32. Furthermore, comparison of accuracy curves and losses of training and testing of CNN+PSO models can be observed in Figures 33 and 34. It can be seen that the loss curve in the CNN+PSO model shows suboptimal results, which may be caused by a decrease in accuracy at the validation stage that causes overfitting.

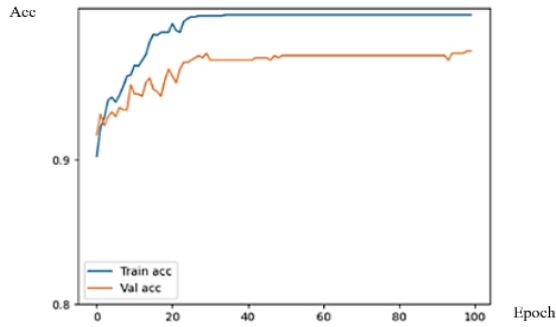


Figure 31. CNN accuracy curve

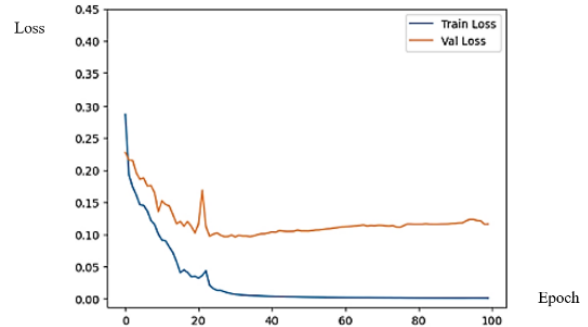


Figure 32. CNN loss curve

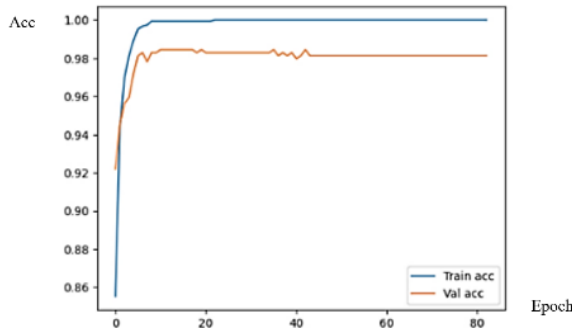


Figure 33. CNN+PSO accuracy curve

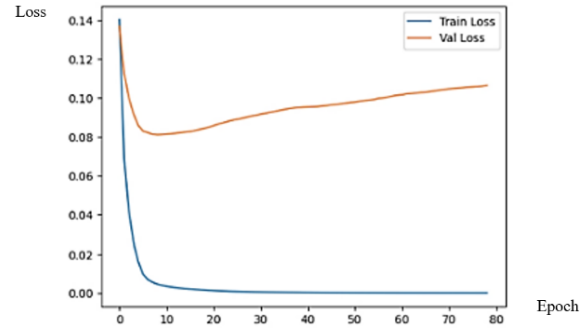


Figure 34. CNN+PSO loss curve

3.9. Model 9

In the ninth model, the ratio of datasets used is 60:40. The training accuracy of the CNN model reached 98.12%. After normalizing the data, the model was trained with the PSO optimization algorithm involved, but this did not result in a significant increase in accuracy, only about 98.01%. Details about the accuracy and loss curves of the CNN model are seen in Figures 35 and 36. The results of the CNN+PSO model also illustrate a similar situation, where the accuracy curve at the validation stage decreases compared to the training accuracy, as a result of the validation loss value continues to increase. This information is shown in Figures 37 and 38.

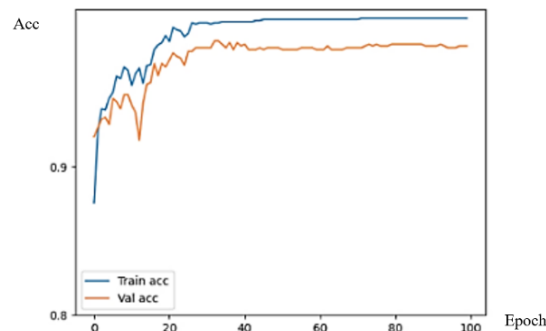


Figure 35. CNN accuracy curve

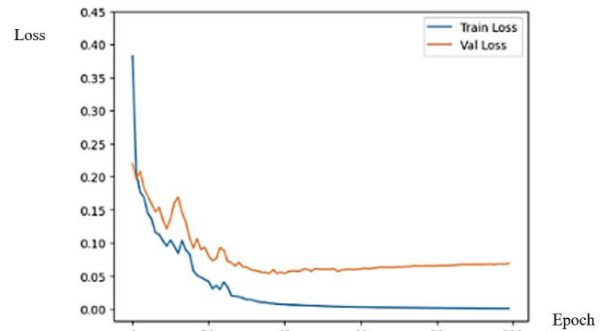


Figure 36. CNN loss curve

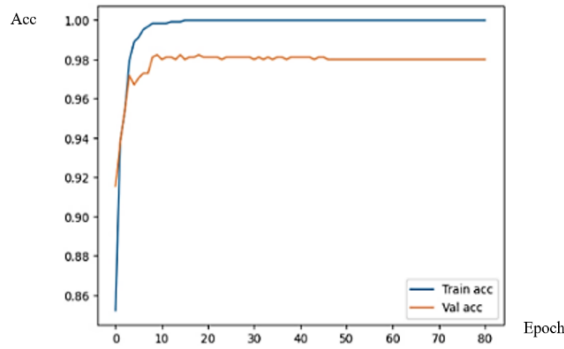


Figure 37. CNN+PSO accuracy curve

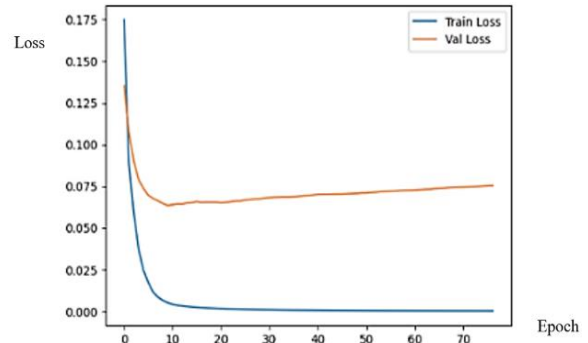


Figure 38. CNN+PSO loss curve

3.10. Model 10

The tenth model, which was the last model to be trained and tested using 50:50 data sharing from the eeg-emotion-2 dataset, achieved an accuracy of 97.28%. As with previous models, it also faces challenges in the form of suboptimal performance curves. A decrease in validation accuracy and an increase in validation loss value also affect the performance of this model.

Information about the accuracy and loss curves of the CNN model is shown in Figures 39 and 40. Similarly, the accuracy curve shown in Figure 41 and loss in the CNN+PSO model reflect a similar situation, almost identical to the fifth model tested with the same data sharing. In both cases, there was an increase in validation loss which resulted in an overfitting tendency as shown in Figure 42.

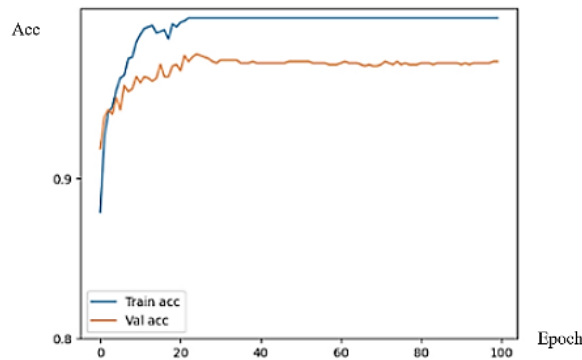


Figure 39. CNN accuracy curve

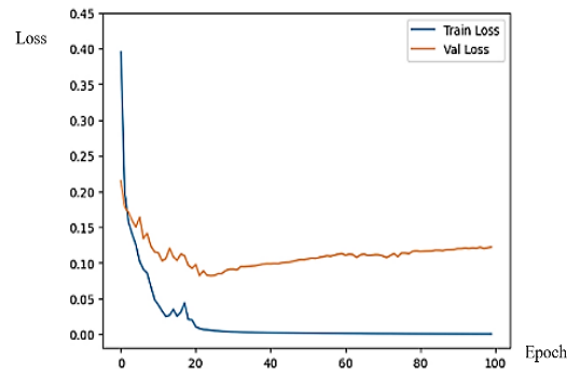


Figure 40. CNN loss curve

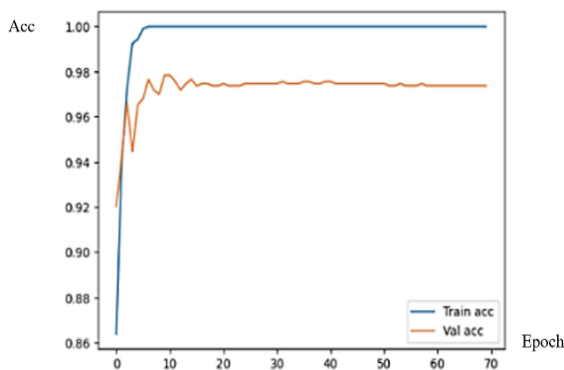


Figure 41. CNN+PSO accuracy curve

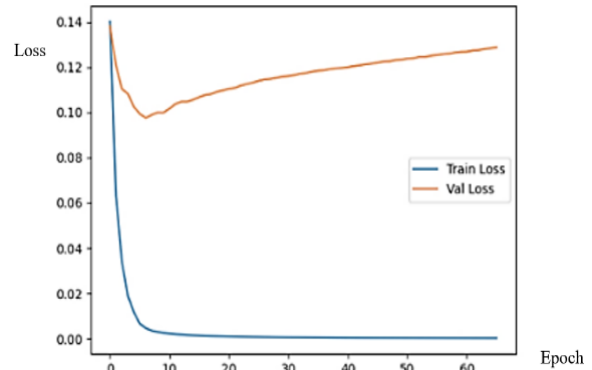


Figure 42. CNN+PSO loss curve

Results from all ten CNN and CNN+PSO models that have been trained and tested, documented and compared in Table 4. In the case of the eeg-emotion-1 dataset, the first model with 90:10 data sharing using CNN obtained the highest accuracy of 99.07%. Meanwhile, training and testing using the eeg-emotion-2 dataset, the seventh model involving the PSO algorithm, achieved a peak accuracy of 99.30%. However, when evaluating the quality of the performance curve, models trained and tested using CNN produced better curves.

Table 4. Comparison of accuracy results

Model	Metode	Accuracy (%)
1	CNN	99.07
	CNN+PSO	98.60
2	CNN	97.19
	CNN+PSO	98.36
3	CNN	97.03
	CNN+PSO	97.19
4	CNN	97.30
	CNN+PSO	97.66
5	CNN	96.90
	CNN+PSO	97.56
6	CNN	97.20
	CNN+PSO	97.66
7	CNN	97.42
	CNN+PSO	99.30
8	CNN	97.50
	CNN+PSO	98.12
9	CNN	98.12
	CNN+PSO	98.01
10	CNN	97.28
	CNN+PSO	97.37

We compared our model with two other models which also use eeg based brainwave dataset. The first model explores single and ensemble methods for classifying emotions by reducing them to smaller datasets through feature selection using OneR scores, Bayes network, information gain, and symmetrical uncertainty. They achieved an overall accuracy of about 97.89% [19] and the second model used an applied evolutionary algorithm to select the most informative features from the initial feature set. Multilayer perceptron (MLP) optimization is performed using an evolutionary approach before classification to estimate the best hyperparameters of the network. Deep learning and adaptation using long short-term memory (LSTM) are also explored, and the adaptive boosting of both types of models is examined for each problem. The obtained results show that an adaptive boosted LSTM can achieve an accuracy of 84.44%, 97.06%, and 9.94% attention, emotion, and numerical datasets, respectively [26]. Compared with the existing techniques, the proposed techniques achieved a higher level of accuracy of 98% by applying the hybrid CNN with PSO.

4. CONCLUSION

The main focus in this study was to classify EEG signal data for emotion detection using CNN algorithms. To enhanced the CNN performance, the hyper parameters in the CNN algorithm are optimized with the PSO algorithm. Based on the evaluation results in each model, hybrid CNN-PSO showed better results an accuracy rate of 99.30% compared to CNN. The comparison between the existing algorithms that used the same data (i.e OneR, Bayes network, information gain, and an adaptive boosted LSTM) shows that hybrid CNN-PSO has better performance. However, although hybrid CNN PSO provides better results, the quality of the performance curve CNN produces a better result. So, for further research is still needed to improve the performance of machine learning by tuning the hyper parameter or using another optimization algorithm.




REFERENCES

- [1] J. Yang, X. Huang, H. Wu, and X. Yang, "EEG-based emotion classification based on bidirectional long short-term memory network," *Procedia Computer Science*, vol. 174, pp. 491–504, 2020, doi: 10.1016/j.procs.2020.06.117.
- [2] S. M. Alarcao and M. J. Fonseca, "Emotions recognition using EEG signals: a survey," *IEEE Transactions on Affective Computing*, vol. 10, no. 3, pp. 374–393, 2019, doi: 10.1109/TAFFC.2017.2714671.
- [3] A. Subasi and M. I. Gursoy, "EEG signal classification using PCA, ICA, LDA and support vector machines," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8659–8666, 2010, doi: 10.1016/j.eswa.2010.06.065.
- [4] Mustaqeem and S. Kwon, "A CNN-assisted enhanced audio signal processing for speech emotion recognition," *Sensors*, vol. 20, no. 1, 2020, doi: 10.3390/s20010183.




- [5] F. E. F. Junior and G. G. Yen, "Particle swarm optimization of deep neural networks architectures for image classification," *Swarm and Evolutionary Computation*, vol. 49, pp. 62–74, 2019, doi: 10.1016/j.swevo.2019.05.010.
- [6] Z. Wen, R. Xu, and J. Du, "A novel convolutional neural networks for emotion recognition based on EEG signal," *2017 International Conference on Security, Pattern Analysis, and Cybernetics, SPAC 2017*, vol. 2018, pp. 672–677, 2017, doi: 10.1109/SPAC.2017.8304360.
- [7] S. B. Sulistyono, W. L. Woo, and S. S. Dlay, "Regularized neural networks fusion and genetic algorithm based on-field nitrogen status estimation of wheat plants," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 103–114, 2017, doi: 10.1109/TII.2016.2628439.
- [8] M. Zhou *et al.*, "Epileptic seizure detection based on EEG signals and CNN," *Frontiers in Neuroinformatics*, vol. 12, 2018, doi: 10.3389/fninf.2018.00095.
- [9] T. Yamasaki, T. Honma, and K. Aizawa, "Efficient optimization of convolutional neural networks using particle swarm optimization," *Proceedings - 2017 IEEE 3rd International Conference on Multimedia Big Data, BigMM 2017*, pp. 70–73, 2017, doi: 10.1109/BigMM.2017.69.
- [10] A. Darwish, D. Ezzat, and A. E. Hassanien, "An optimized model based on convolutional neural networks and orthogonal learning particle swarm optimization algorithm for plant diseases diagnosis," *Swarm and Evolutionary Computation*, vol. 52, 2020, doi: 10.1016/j.swevo.2019.100616.
- [11] N. M. Aszemi and P. D. D. Dominic, "Hyperparameter optimization in convolutional neural network using genetic algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 6, pp. 269–278, 2019, doi: 10.14569/ijacsa.2019.0100638.
- [12] V. Passricha and R. K. Aggarwal, "PSO-based optimized CNN for Hindi ASR," *International Journal of Speech Technology*, vol. 22, no. 4, pp. 1123–1133, 2019, doi: 10.1007/s10772-019-09652-3.
- [13] Y. Wang, H. Zhang, and G. Zhang, "cPSO-CNN: an efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks," *Swarm and Evolutionary Computation*, vol. 49, pp. 114–123, 2019, doi: 10.1016/j.swevo.2019.06.002.
- [14] G. Silva, T. Valente, A. Silva, A. Paiva, and M. Gattass, "Convolutional neural network-based PSO for lung nodule false positive reduction on CT images," *Computer Methods and Programs in Biomedicine*, vol. 162, pp. 109–118, 2018, doi: 10.1016/j.cmpb.2018.05.006.
- [15] M. Marhatang and R. D. Muhammad, "Optimal economic dispatch using particle swarm optimization in Sulselbar system," *IAES International Journal of Artificial Intelligence*, vol. 11, no. 1, pp. 221–228, 2022, doi: 10.11591/ijai.v11.i1.pp221-228.
- [16] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: an overview," *Soft Computing*, vol. 22, pp. 387–408, 2018, doi: 10.1007/s00500-016-2474-6.
- [17] P. Singh, S. Chaudhury, and B. K. Panigrahi, "Hybrid MPSO-CNN: multi-level particle swarm optimized hyperparameters of convolutional neural network," *Swarm and Evolutionary Computation*, vol. 63, pp. 1–13, Jun. 2021, doi: 10.1016/j.swevo.2021.100863.
- [18] J. J. Bird, L. J. Manso, E. P. Ribeiro, A. Ekart, and D. R. Faria, "A study on mental state classification using EEG-based brain-machine interface," *9th International Conference on Intelligent Systems 2018: Theory, Research and Innovation in Applications, IS 2018 - Proceedings*, pp. 795–800, 2018, doi: 10.1109/IS.2018.8710576.
- [19] J. J. Bird, A. Ekart, C. Buckingham, and D. R. Faria, "Mental emotional sentiment classification with an EEG-based brain-machine interface," *Proceedings of the International Conference on Digital Image and Signal Processing (DISP'19)*, Oxford University, UK, pp. 1–7, 2019.
- [20] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *Journal of Big Data*, vol. 7, no. 1, 2020, doi: 10.1186/s40537-020-00305-w.
- [21] J. Lanchantin, T. Wang, V. Ordonez, and Y. Qi, "General multi-label image classification with transformers," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 16473–16483, 2021, doi: 10.1109/CVPR46437.2021.01621.
- [22] V. A. Yulianto, N. Effendy, and A. Arif, "Finger vein identification system using capsule networks with hyperparameter tuning," *IAES International Journal of Artificial Intelligence*, vol. 12, no. 4, pp. 1636–1643, Dec. 2023, doi: 10.11591/ijai.v12.i4.pp1636-1643.
- [23] S. Cebollada, L. Payá, M. Flores, A. Peidró, and O. Reinoso, "A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data," *Expert Systems with Applications*, vol. 167, 2021, doi: 10.1016/j.eswa.2020.114195.
- [24] L. Xie, J. Tao, Q. Zhang, and H. Zhou, "CNN and KPCA-based automated feature extraction for real time driving pattern recognition," *IEEE Access*, vol. 7, pp. 123765–123775, 2019, doi: 10.1109/ACCESS.2019.2938768.
- [25] A. A. Barbhuiya, R. K. Karsh, and R. Jain, "CNN based feature extraction and classification for sign language," *Multimedia Tools and Applications*, vol. 80, no. 2, pp. 3051–3069, 2021, doi: 10.1007/s11042-020-09829-y.
- [26] J. Bird, D. Faria, L. Manso, A. Ekart, and C. Buckingham, "A deep evolutionary approach to bioinspired classifier optimisation for brain-machine interaction," *Complexity*, vol. 2019, 2019, doi: 10.1155/2019/4316548.

BIOGRAPHIES OF AUTHORS






Dian Palupi Rini    holds a Doctor of Computer Science degree from University Technology Malaysia, in 2017. She also received her B.Sc. (Mathematics) from Sriwijaya University, Indonesia, and M.Sc. (Computer Science) from Gadjahmada University, Indonesia in 2000 and 2003, respectively. She is currently a lecturer at Computer Science Faculty, Sriwijaya University, Indonesia. Her research includes meta-heuristics, global optimization, machine learning, deep learning, data mining, fuzzy systems, and artificial neural network. She can be contacted at email: dprini@unsri.ac.id.






Tri Kurnia Sari    holds the Bachelor of Informatic degree (S.Kom.) from Sriwijaya University in 2019 and also Master of Computer Science (M.Kom.) from Sriwijaya University in 2023. She is currently a civil servant as an information technology staff in Padang Selasa Public Health Center at Palembang City. Her research interest includes data mining. She can be contacted at email: trikurniasari11@gmail.com.



Winda Kurnia Sari    earned a Bachelor's degree in Information Systems (S.SI.) with a focus on knowledge management, and a Master's degree in Computer Science (M.Kom.) specializing in Business Intelligence and Natural Language Processing. Currently, she teaches in the Department of Information Systems at the Faculty of Computer Science, Universitas Sriwijaya. Her research interests include natural language processing, machine learning, and deep learning. She can be contacted at email: windakurniasari@unsri.ac.id.



Novi Yusliani    is currently a lecturer and researcher in Faculty of Computer Science, Universitas Sriwijaya, Indonesia. Her research interest includes natural language processing, deep learning, and machine learning. She can be contacted at email: novi_yusliani@unsri.ac.id.