Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

Hybrid Neural Network-Based Road Damage Detection Using CNN-RNN and CNN-MLP Models

Ani Dijah Rahajoe*[1], Muhammad Suriansyah[2], Angelo A. Beltran Jr[3]

[1]Computer Science, Universitas Pembangunan Nasional Veteran Jawa Timur, Indonesia
[2]Postgraduate, Universitas Diponegoro, Semarang, Indonesia
[3] Electronics Engineering, Adamson University, Manila, Philipines

Email: [1]anidijah.if@upnjatim.ac.id

Received : Feb 17, 2025; Revised : May 11, 2025; Accepted : May 27, 2025; Published : Jun 10, 2025

Abstract

Currently, there are many applications of image processing in various fields. One of them is the recognition of paved road images. Detection through images helps in handling infrastructure development roads. With the advancement of technology, especially in the field of deep learning, the process of detecting road damage can be done automatically and more efficiently. The road damage detection system can be integrated into the smart city system to monitor infrastructure conditions in real time. This study will use a combined deep learning algorithm between Convolutional Neural Network- Recurrent Neural Network (CNN-RNN) and as a comparison using Convolutional Neural Network-MultiLayer Perceptrons (CNN-MLP). The study aims to analyze the accuracy of using the CNN-RNN and CNN-MLP algorithms for detecting paved roads that have categories of undamaged roads, damaged roads, and damaged roads with holes. The detection of paved roads has complex details so an algorithm that has good performance with high accuracy is needed. The results of the study showed that the CNN-RNN hybrid had a better accuracy of 96.59 percent than the CNN-MLP hybrid model of 95.9 percent.

Keywords : *Convolutional Neural Network, Damaged Road Detection, MultiLayer Perceptron, Recurrent Neural Network.*

This work is an open access article and licensed under a Creative Commons Attribution-Non Commercial 4.0 International License

1. INTRODUCTION

This study discusses road damage detection. Damage detection is an important step in road infrastructure maintenance. The tradition of road damage detection is often done manually, which requires a lot of time and effort. With the advancement of technology, especially in the field of deep learning, the road damage detection process can be done automatically and more efficiently. The most important stage is when taking pictures of damaged roads. This is related to the angle and direction of the image capture. The more the angle of the image capture is not appropriate, the accuracy obtained will not be optimal.

Road infrastructure maintenance through accurate and up-to-date road damage detection, the government or related institutions can plan maintenance schedules more efficiently. Priority repairs can be given to the most severely damaged road sections quickly and accurately. Early detection of damage can prevent more severe and expensive damage to repair. Road damage that is not immediately repaired can endanger road users. Early detection and rapid repair can reduce the risk of accidents. Detection of road damage in the city is in line with the development of a city. Smooth and safe roads improve the quality of life of city residents. Reducing congestion and pollution due to road damage also contributes to the quality of life. Good road infrastructure supports economic activities, such as the transportation of goods and services. Investment becomes more attractive when city infrastructure, including roads, is in good condition. The use of technologies such as deep learning can support efforts towards a

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

sustainable smart city. The road damage detection system can be integrated into the smart city system to monitor infrastructure conditions in real time [1][2]. This study will use the CNN-RNN deep learning algorithm and as a comparison using CNN-MLP.

Hybrid CNN-RNN is an artificial neural network architecture that combines the power of Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). This algorithm is included in deep learning [3]. The difference between machine learning and deep learning lies in the learning process. In machine learning through the preprocessing process, feature extraction, feature selection, and classification [3]. The Deep learning method goes directly to the deep learning model process and then to the classification stage. If machine learning is widely used in the fields of text mining, spam detection, image classification, video, and multimedia, then deep learning is widely used in big data analysis, cyber security, medical informatics, social media, radiology, and cancer detection [4][5][6][7][8]. Deep learning research for stock time series data was also conducted by [9]. Accident classification included in the Intelligent Transportation System (ITS) has been studied by [10] using CNN and RNN. This algorithm has also been applied to smart home application research [11]. Even for music classification [12]. The CNN-RNN combination is also applied to rainfall-runoff classification [13]. CNN-RNN combines CNN's ability to extract spatial features with RNN's ability to process sequential information. As a comparison, this study also uses CNN-MLP (Multilayer Perceptrons). CNN is very effective in extracting spatial features from image data, while MLP is used in modeling non-linear relationships between these features. CNN-MLP can produce models for various tasks such as image classification, segmentation, and object detection [3][14]. The classification results have 3 labels, namely the category of undamaged roads, damaged roads, and damaged roads with holes.

The first contribution of this research is the analysis of the combined combination of CNN-MLP and CNN-RNN algorithms in the process of detecting paved road images. As explained in the previous paragraph, paved roads have 3 labels, where the detection between damaged roads but not potholed and damaged roads with potholed roads has a complex detection. Thus, a good algorithm is needed to be able to classify with high accuracy. This study shows that CNN-RNN has a better accuracy of 96.59 percent than the CNN-MLP model of 95.9 percent. This study consists of Chapter 1, namely the introduction, chapter 2 is the research method used, chapter 3 is the result and discussion section and Chapter 4 is the conclusion section.

2. METHOD

Compared to previous methods, CNN offers significant advantages in detecting features independently. This makes CNN the most widely used method. CNN automatically identifies relevant features without human supervision [15].

2.1. Convolutional Neural Network (CNN)

The structure of CNN is described as a human brain neuron similar to a conventional neural network. Complex cell circuits are simulated by [16]. The common CNN architecture consists of a stack of recurrent convolutional layers, followed by a pooling layer to reduce the dimensionality of the data. The fully-connected (FC) layer at the end of the network functions to perform classification. The input to each CNN layer is a three-dimensional tensor with size m x m x r. Dimension m represents height and width, while r represents the number of channels. The convolution filter is also a three-dimensional tensor with size n x n x q, where n is smaller than m.

The requirements $n < m$ and $q \leq r$ must be met. The kernel, as the basic unit of local connectivity, has shared weights in the form of bias b^k and weight W^k to produce k feature maps h^k of size (m-n-1). These feature maps are then convolved with the input data. The next step is to apply a nonlinear activation function to the output of the convolution layer as in the equation 1 [3]:

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

$$h^k = f\left(W^k * x + b^k\right) \quad (1)$$

The sub-sampling layer samples each feature map reduces network parameters, and speeds up training. This helps prevent overfitting. A pooling function (such as max or mean) is applied to the p × p area of each feature map. The final fully-connected (FC) layer combines mid-level and low-level features to create a high-level abstraction, similar to a traditional neural network.

2.2. Recurrent Neural Network (RNN)

This is an example of the use of sub-chapters in a paper. Sub-chapters are allowed to be included in all chapters, except in the conclusion. RNN recursively processes each element in a sequence, with the output from the previous time step being used as input for the next time step. The RNN architecture consists of interconnected layers, allowing for a bidirectional flow of information [17][18][19]. Each neuron in the RNN performs both linear and non-linear computations to produce the output. The weights in the network are learned through a backpropagation algorithm, which allows the RNN to model temporal relationships in the data [19][20].

RNNs are specifically designed to process data that has sequences. By using feedback connections, RNNs can retain information from the past and use it to predict the future. The hidden state in an RNN is a vector that stores information about previous inputs as shown in the equation 2 [21][22].

$$S_t = tanh\left(w_{x_t} + US_{t-1} + b\right) \quad (2)$$

$$o_t = c + VS_t \quad (3)$$

Where the input vector at time t is x_t. The current state vector at time t is h_t. While the symbol U is the hidden-to-hidden weight matrix and the input-to-hidden weight matrix is W. V is the hidden-to-output weight matrix. The bias term is symbolized by b and c. A simple RNN diagram can be seen in [22][23].

2.3. Multilayer Perceptron (MLP)

MLP consists of sev[23]eral layers (input, hidden, and output) that are interconnected. Each layer is fully connected to the previous and next layers. The MLP architecture is simulated by Daniel Svozil et al. [24][25] as neurons that work in the human brain. Each node in the MLP produces output by combining weights and input values, then processed with a nonlinear activation function to separate complex data as in the equation 4 [26][27].

$$a^{(l+1)} = \sigma(w^1 a^1 + b^1) \quad (4)$$

The variable l represents the layer index. The weights and biases at the first layer are denoted as $w^{(l)}$ and $b^{(l)}$. A nonlinear activation function σ (e.g., sigmoid, hyperbolic tangent, or ReLU) is used to introduce nonlinearity at the output of the first layer. For a multilayer neural network with m layers, the activation at the first input layer is $a^{(l)} = x$, and the activation at the last output layer is as shown in the equation 5.

$$h_{w,b}(x) = a^{(m)} \quad (5)$$

The weight parameters (w) and bias (b) in equation 5 are learned iteratively through the backpropagation algorithm to map the complex input-output relationships in the data. The objective function used is the squared error, which measures the difference between the predicted output and the target output as shown in the equation 6 [26].

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

$$J(W,b;x,y) = \frac{1}{2}\|h_{w,b}(x) - y\|^2 \qquad (6)$$

The MLP and CNN predict n-dimensional probability vectors $C=(C_1, C_2,,\ldots,C_n)$ for each pixel, where n is the number of classes. Each dimension of $i \in [1,n]$ represents the probability of the pixel belonging to the corresponding class. The predicted probability for the correct class would be one, while others would be 0. However, due to uncertainties in remote sensing, the actual probabilities are denoted as $f(x) = \{c_x \mid x \in (1,2,\ldots,n)\}$, where $c_x \in [0,1]$ and $\sum_1^n c_x = 1$. The classification model selects the class with the highest probability as the predicted output in the equation 7 [26].

$$class(C) = argmax(\{x \in (1,2,\ldots,n)\}) \qquad (7)$$

The method proposed in this study is smart road detection taken from a smartphone camera to determine whether the road is cracked, cracked, potholed, or not damaged (not potholed and not cracked) using deep learning, a combination of Convolutional Neural Network (CNN) - Simple RNN. For comparison, CNN-MLP is used. Smart detection uses an Android application to make it easier for users to take pictures and display the results of the road detection. The methodology of this study can be seen in Figure 1. Deep learning is taken from the dataset on Kaggle (more in the dataset section) using CNN as a feature extraction in the layer, then the convolution layer and ReLU (Rectified Linear Unit) process are carried out, followed by a pooling layer and repeated up to n. ReLu is an activation function commonly used in artificial neural networks, especially in deep learning. This function maps all negative values to 0, while positive values are left unchanged. Furthermore, the process produces flattening [25]. Flatten is a process in artificial neural networks, especially in the Convolutional Neural Network (CNN) architecture, which functions to convert multi-dimensional data into one dimension. This multi-dimensional data is usually the output of the convolution and pooling layers in the form of tensors (multi-dimensional arrays). Flatten allows us to connect the CNN feature extraction section with the RNN classification section. RNN will process the feature vector generated by CNN to determine the class of an object. The flattening process is very simple, namely changing all elements in the tensor into one long row. The results of this flattening become input for classification using RNN. RNN in this research predicts 3 labels, namely undamaged roads, cracked roads, and cracked roads with holes. The activation function in this study uses softmax. ReLU (Rectified Linear Unit) is an activation function commonly used in artificial neural networks, especially in deep learning.

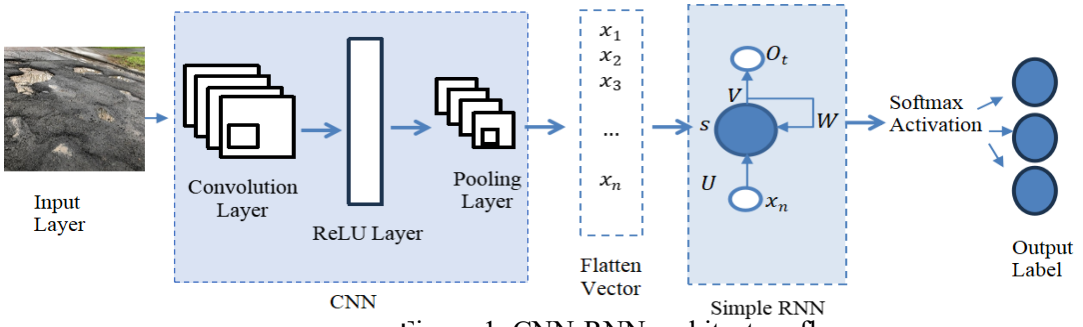

Figure 1. CNN-RNN architecture flow

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

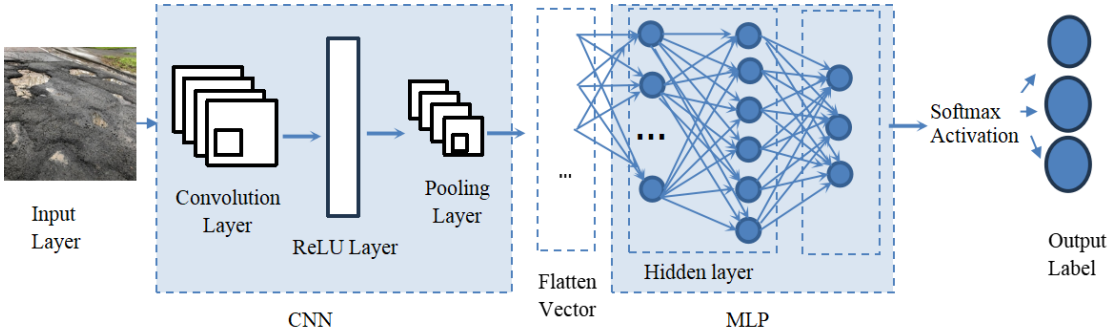

Figure 2. CNN-MLP Architecture Flow

a. Road image not damaged b. Cracked Road c. Cracked potholed road

Figure 3. Types of roads detected in this research

Softmax takes the input vector from the neurons in the previous layer and converts it into a new probability vector. Each value in the output vector generated by softmax represents the probability that the input belongs to a particular class. The softmax activation function is as in the equation 8.

$$softmax(z\_i) = exp\ exp\ (z\_i) / \sum\ \ exp\ (z\_i) \qquad (8)$$

Where z_i is the output value of the first neuron in the system before softmax and $\sum \exp\ (z\_i)$ is the exponential sum of all output values. The proposed CNN-RNN methodology flow can be seen in Figure 1 and CNN-MLP in Figure 2.

In this research, tensorFlow Lite was used which was integrated into an Android application as was done in previous studies [28].

2.4. Dataset

Deep learning requires a more precise and quality dataset to approach maximum accuracy. The training data is taken from the Kaggle dataset link https://www.kaggle.com/datasets/rosyid/indonesia-crack-road. Figure 3 shows an example of a road in the dataset. In this study, the images in the dataset used include various road conditions, namely undamaged roads (Figure 3.a), cracked roads (Figure 3.b), and cracked roads with holes (Figure 3.c). The total data is 734. The training testing data uses train_test_split from sci-kit-learn with a proportion of training data of 90%, 80%, and 70% and testing data of 10%, 20%, and 30%. The highest accuracy results will be used for the input data model from the camera.

3. RESULT

The images in the dataset used as training data have been labeled. The preprocessing is converted to grayscale and then the pixel dimensions are reduced to 100 x 100. Furthermore, the labeling is changed to numbers 0, 1, and 2. Number 0 for the label of undamaged roads, 1 for cracked roads, and 2 for cracked roads without holes. Normalization is used for image pixels 255.0 for a scale of 0-255 to 0-1. The kernel used is 3 x 3 and uses MaxPooling2D. Its activation function for multiclass uses Softmax.

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

In this study, the Adam (Adaptive Moment Estimation) optimizer is used to iteratively update the weights in the Neural Network. The model can learn from data and will be more accurate in making predictions [29]. This algorithm adaptively adjusts the learning rate for each model parameter. The loss function uses Sparse Categorical Crossentropy for multiclass classification.

Table 1. Training Models

Neural Networks	Layers	Output Models	Number of Parameters
CNN-RNN	conv2d (Conv2D)	(None, 98, 98, 32)	320
	max_pooling2d (MaxPooling2D)	(None, 49, 49, 32)	0
	conv2d_1 (Conv2D)	(None, 47, 47, 64)	18,496
	max_pooling2d_1 (MaxPooling2D)	(None, 23, 23, 64)	0
	conv2d_2 (Conv2D)	(None, 21, 21, 128)	73,856
	max_pooling2d_2 (MaxPooling2D)	(None, 10, 10, 128)	0
	flatten (Flatten)	(None, 12800)	0
	reshape (Reshape)	(None, 1, 12800)	0
	simple_rnn (SimpleRNN)	(None, 50)	642,550
	dense (Dense)	(None, 3)	153
CNN-MLP	conv2d (Conv2D)	(None, 98, 98, 32)	320
	max_pooling2d (MaxPooling2D)	(None, 49, 49, 32)	0
	conv2d_1 (Conv2D)	(None, 47, 47, 64)	18,496
	max_pooling2d_1 (MaxPooling2D)	(None, 23, 23, 64)	0
	conv2d_2 (Conv2D)	(None, 21, 21, 128)	73,856
	max_pooling2d_2 (MaxPooling2D)	(None, 10, 10, 128)	0
	flatten (Flatten)	(None, 12800)	0
	reshape (Reshape)	(None, 1, 12800)	0
	MLP	(None, 256)	3,277,056
	dense_1 (Dense)	(None, 128)	32,896
	dense_2 (Dense)	(None, 3)	387

Table 2. CNN-RNN and CNN-MLP accuracy results

Training Data	Testing Data	CNN - RNN		CNN – MLP	
		Accuracy	Accuracy percentage	Accuracy	Accuracy percentage
90%	10%	0.9591	95,91%	0.9455	94,55%
80%	20%	0.9659	96,59%	0.9591	95,91%
70%	30%	0.9455	94,55%	0.9252	92,52%

3.1. Training Model

Training uses 100 epochs and a batch size of 32 samples per update. The CNN layer goes through 3 convolution layers which are continued at the pooling layer stage to reduce the data dimension from 98x98x32 to 10x10x128. The dimension reduction stages can be seen in table 1.

Furthermore, the flatten process is carried out to become one dimension as input to RNN and MLP. In CNN-RNN, the flatten process becomes 12800 outputs and is reshaped to 1.12800. After using Simple RNN, it becomes 50 outputs and using softmax to 3 from 153 parameters. CNN-MLP requires dense twice to become 3 labels as output.

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

The results of the data evaluation are shown in Table 2. In the three training and test data scenarios, it can be seen that CNN-RNN has higher accuracy than CNN-MLP. The highest accuracy is found in CNN-RNN with 96.59 per cent for the training data scenario of 80 percent and test data of 20 percent. While in CNN-MLP the highest accuracy is 95.91 percent for the training data scenario of 80 percent and test data of 20 percent. This shows that the 80 per cent training data scenario is the basis for the formation of the next classification model.

3.2. Loss and accuracy values for CNN-RNN

The smaller the loss value, the better the model is at making predictions. At the beginning of training, both training and testing losses tend to be high at the beginning of training. This indicates that the model still does not understand the patterns in the data. During training, the training loss generally decreases significantly as the epoch progresses. Figure 4 shows the testing accuracy of CNN-RNN and Figure 5 for CNN-MLP in the scenario of 80 percent training data and 20 percent test data.

```
Epoch 92/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 9.3613e-05 - val_accuracy: 0.9660 - val_loss: 0.1454
Epoch 93/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 1.1365e-04 - val_accuracy: 0.9660 - val_loss: 0.1372
Epoch 94/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 8.5631e-05 - val_accuracy: 0.9660 - val_loss: 0.1494
Epoch 95/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 7.6452e-05 - val_accuracy: 0.9660 - val_loss: 0.1617
Epoch 96/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 8.6066e-05 - val_accuracy: 0.9660 - val_loss: 0.1528
Epoch 97/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 6.5019e-05 - val_accuracy: 0.9660 - val_loss: 0.1667
Epoch 98/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 1.0000 - loss: 7.2886e-05 - val_accuracy: 0.9660 - val_loss: 0.1531
Epoch 99/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 1.0000 - loss: 7.7105e-05 - val_accuracy: 0.9660 - val_loss: 0.1644
Epoch 100/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 1.0000 - loss: 7.7889e-05 - val_accuracy: 0.9660 - val_loss: 0.2107
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9583 - loss: 0.2868
Test accuracy: 0.9659863710403442
```

Figure 4. CNN-RNN testing accuracy

```
Epoch 92/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.2536e-05 - val_accuracy: 0.9660 - val_loss: 0.1030
Epoch 93/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.3060e-05 - val_accuracy: 0.9660 - val_loss: 0.1030
Epoch 94/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.8330e-05 - val_accuracy: 0.9660 - val_loss: 0.1041
Epoch 95/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.3413e-05 - val_accuracy: 0.9660 - val_loss: 0.1024
Epoch 96/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.0372e-05 - val_accuracy: 0.9660 - val_loss: 0.1018
Epoch 97/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 1.0000 - loss: 2.2536e-05 - val_accuracy: 0.9660 - val_loss: 0.1010
Epoch 98/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 1.0000 - loss: 2.2945e-05 - val_accuracy: 0.9660 - val_loss: 0.1058
Epoch 99/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 1.8963e-05 - val_accuracy: 0.9660 - val_loss: 0.1003
Epoch 100/100
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 1.0000 - loss: 2.0065e-05 - val_accuracy: 0.9592 - val_loss: 0.1050
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9573 - loss: 0.1045
Test accuracy: 0.9591836929321289
```

Figure 5. CNN-MLP testing accuracy

This indicates that the model is getting better at predicting training data. The testing loss also decreases, but at a slower rate and with less fluctuation compared to the training loss. This indicates that the model is learning common features that can be generalized to new data. Accuracy in CNN-RNN is a metric used to measure how often our model makes correct predictions. Accuracy values range from 0 to 1, where 1 indicates perfect accuracy. Both training and testing accuracy tend to be low at the beginning of training. This indicates that the model still does not understand the patterns in the data.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

During training, the training accuracy generally increases significantly as the epoch progresses. This indicates that the model is getting better at predicting training data.

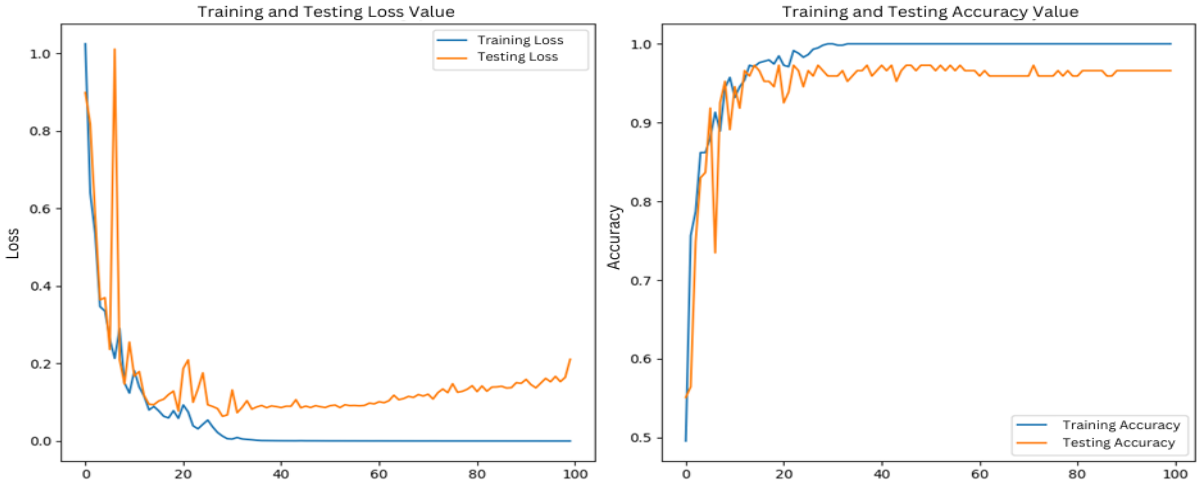

Figure 6. Loss and accuracy value graph for CNN-RNN

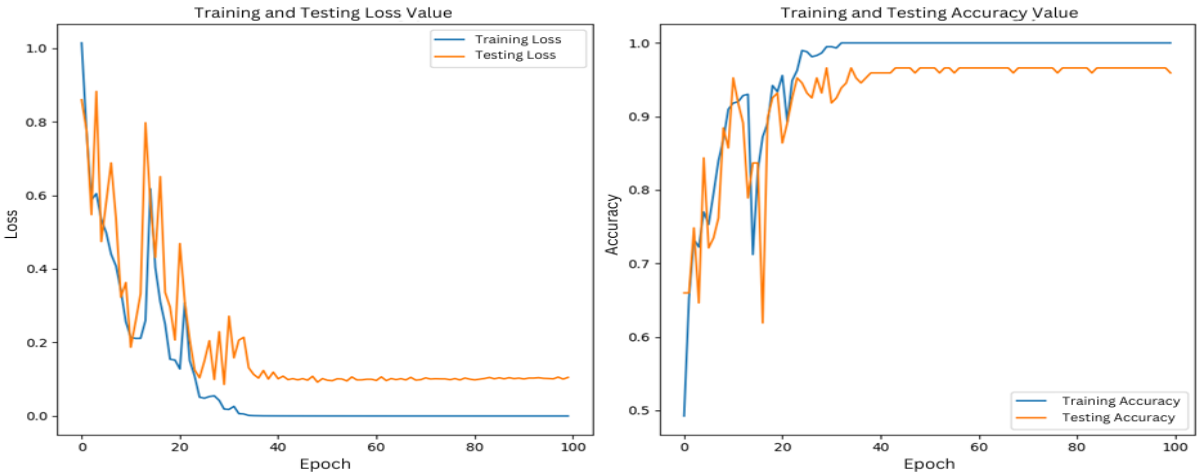

Figure 7. Loss and accuracy value graph for CNN-MLP

Testing accuracy also increases, but at a slower rate and with less fluctuation compared to training accuracy. This indicates that the model is learning common features that can be generalized to new data. At epoch 90 the testing accuracy remains constant, indicating that the model has reached maximum accuracy.

3.3. Loss value and accuracy of CNN-MLP

In Figure 7, the initial training loss and testing loss tend to be high. This indicates that the model still does not understand the patterns in the data. The blue curve shows the average loss value when the model is trained on the training data. The lower the loss value, the better the model is at fitting the training data. The orange curve shows the average loss value when the model is evaluated on previously unseen testing data. This gives an indication of how well the model can generalize to new data. Training and testing accuracy tend to be low at the beginning of training. Training accuracy increases significantly as the number of epochs increases, following the decreasing trend of training loss. Testing accuracy also increases and reaches a plateau after a few epochs. This indicates that the model has achieved its best performance on the testing data.

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

4. DISCUSSIONS

Figure 6 also shows the accuracy curve graph for training data and testing data during the epoch. The training accuracy curve line increases sharply at epoch 30 to approach a value of 1 or 100 percent. In Figure 6, the model successfully captures complex patterns in the training data, as indicated by a significant decrease in training loss. The CNN-RNN model is able to make accurate predictions on previously unseen testing data, as indicated by the high and stable testing accuracy. The graph in Figure 6 shows that the model does not experience serious overfitting, because there is no significant increase in testing loss after reaching the lowest point. This means that the model does not memorize the training data too much and can adapt to new data.

In Figure 7, the CNN-MLP model shows very good performance. Both loss and accuracy on the testing data reach stable and high values, indicating that the model has successfully learned important features of the data and can generalize well. The graph in Figure 7 shows that the model does not experience serious overfitting, because there is no significant increase in testing loss after reaching the lowest point. This means that the model does not memorize the training data too much and can adapt to new data. Hyperparameter configurations such as learning rate, number of layers, and number of neurons may have been well-tuned to achieve optimal results. Based on the graph 7, it can be concluded that the CNN-MLP model has been trained very well and achieved excellent performance.

5. CONCLUSION

In this research, there are 3 divisions of training data and testing data, namely 70 per cent, 80 percent and 90 percent training data. In the 80 per cent scheme, the testing data has a higher accuracy so that the next classification model is used. This study uses CNN feature extraction which then in the flattened stage results uses simple RNN for classification. As a comparison, CNN-MLP is used where feature extraction also uses CNN but the classification uses MLP. The CNN layer consists of 3 convolutional layers and is followed by a pooling layer. Gradually from dimensions 98x98x32 to 10x10x128. This study shows that CNN-RNN has a better accuracy of 96.59 percent than the CNN-MLP model of 95.9 percent. Although there is a little overfitting in CNN-RNN, the model has recognized new objects well. Further research suggestions use RNN algorithm variants such as Bidirectional Gated Recurrent Unit which can capture 2-way information.

CONFLICT OF INTEREST

The authors declares that there is no conflict of interest between the authors or with research object in this paper.

REFERENCES

[1] E. E. Alahi *et al.*, "Integration of IoT-Enabled Technologies and Artificial Advancements and Future Trends," *Sensors (Switzerland)*, vol. 23, no. 5206, 2023.

[2] A. Ullah *et al.*, "Smart cities: the role of Internet of Things and machine learning in realizing a data-centric smart environment," *Complex and Intelligent Systems*, vol. 10, no. 1, pp. 1607–1637, 2024, doi: 10.1007/s40747-023-01175-4.

[3] L. Alzubaidi *et al.*, *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, vol. 8, no. 1. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.

[4] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, no. November 2016, pp. 11–26, 2017, doi: 10.1016/j.neucom.2016.12.038.

[5] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," *Smart Innovation, Systems and Technologies*, vol. 195, pp. 21–30, 2021, doi: 10.1007/978-981-15-7078-0_3.

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

[6] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Deep Learning applications for COVID-19," *Journal of Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-020-00392-9.
[7] Y. Benhammou, B. Achchab, F. Herrera, and S. Tabik, "BreakHis based breast cancer automatic diagnosis using deep learning: Taxonomy, survey and insights," *Neurocomputing*, vol. 375, pp. 9–24, 2020, doi: 10.1016/j.neucom.2019.09.044.
[8] E. Wulczyn *et al.*, "Deep learning-based survival prediction for multiple cancer types using histopathology images," *PLoS ONE*, vol. 15, no. 6, pp. 1–18, 2020, doi: 10.1371/journal.pone.0233678.
[9] S. Zaheer, N. Anjum, S. Hussain, A. D. Algarni, and J. Iqbal, "A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model," *Mathematical Programming Computation*, vol. 11, pp. 1–24, 2023, doi: https://doi.org/10.3390/ math11030590.
[10] R. Kanakala, J. Mohan, and K. Reddy, "Measurement : Sensors Modelling a deep network using CNN and RNN for accident classification," *Measurement: Sensors*, vol. 27, no. May, p. 100794, 2023, doi: 10.1016/j.measen.2023.100794.
[11] J. Yu, A. de Antonio, and E. Villalba-Mora, "Deep Learning (CNN , RNN) Applications for Smart Homes : A Systematic Review," *computers*, vol. 11, no. Ml, pp. 1–32, 2022, doi: https:// doi.org/10.3390/computers11020026.
[12] M. Ashraf *et al.*, "Hybrid CNN and RNN Variant Model for Music Classification," *applied sciences MDPI*, vol. 13, 2023, doi: https://doi.org/10.3390/ app13031476.
[13] P. Raja, A. Mathew, P. V Yeswanth, and S. Deivalakshmi, "A combined deep CNN-RNN network for rainfall-runoff modelling in Bardha Watershed , India," *Artificial Intelligence in Geosciences*, vol. 5, no. February, p. 100073, 2024, doi: 10.1016/j.aiig.2024.100073.
[14] G. Yao, T. Lei, and J. Zhong, "A review of Convolutional-Neural-Network-based action recognition," *Pattern Recognition Letters*, vol. 118, pp. 14–22, 2019, doi: 10.1016/j.patrec.2018.05.018.
[15] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018, doi: 10.1016/j.patcog.2017.10.013.
[16] D. . Hubel and T. . Wiesel, "RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT ' S VISUAL CORTEX From the Neurophysiolojy Laboratory , Department of Pharmacology central nervous system is the great diversity of its cell types and inter- receptive fields o," *J. physiol*, vol. 160, pp. 106–154, 1962.
[17] S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Feedforward Sequential Memory Networks : A New Structure to Learn Long-term Dependency," *arXiv Prepr. arXiv, 1512 (2015)*, p. 08301, 2015.
[18] A. A. Alsumaiei, "Short-term forecasting of monthly water consumption in hyper-arid climate using recurrent neural networks," *Journal of Engineering Research (Kuwait)*, vol. 9, no. September, pp. 56–69, 2021, doi: https://dx.doi.org/10.36909/jer.v9i3B.10893.
[19] M. Ibrahim and R. Elhafiz, "Modeling an intrusion detection using recurrent neural networks," *Journal of Engineering Research*, vol. 11, no. 1, p. 100013, 2023, doi: 10.1016/j.jer.2023.100013.
[20] G. Bebis and M. Georgiopoulos, "OPTIMAL FEED-FORWARD NEURAL NETWORK ARCHITECTURES," in *IEEE Potentials*, 1994, pp. 27–31.
[21] J. T. Connor, R. D. Martin, L. E. Atlas, and M. Ieee, "Recurrent Neural Networks and Robust Time Series Prediction," in *IEEE Transaction on Neural Networks*, 1994, pp. 240–254.
[22] P. Dey, E. Hossain, I. Hossain, and M. A. Chowdhury, "Comparative Analysis of Recurrent Neural Networks in Stock Price Prediction for Different Frequency Domains," *Algorithms*, vol. 14, pp. 1–20, 2021, doi: https://doi.org/10.3390/a14080251.
[23] F. Mei, H. Chen, and Y. Lei, "Blind Recognition of Forward Error Correction Codes Based on Recurrent Neural Network," *sensors*, vol. 21, 3884, 2021, doi: https://doi.org/10.3390/ s21113884.
[24] Svozil D, V. Kvasnicka, and J. Pospı´chal, "Introduction to multilayer feed-forward neural networks," *Chemom Intell Lab Syst*, vol. 39, no. 1, pp. 43–62, 1997.
[25] C. Do Xuan, "A novel approach for APT attack detection based on combined deep learning

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435

model," *Neural Computing and Applications*, vol. 5, 2021, doi: 10.1007/s00521-021-05952-5.

[26] C. Zhang *et al.*, "A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification," *ISPRS Journal of Photogrammetry and Remote Sensing*, 2017, doi: 10.1016/j.isprsjprs.2017.07.014.

[27] F. Paci, M. Chini, and W. J. Emery, "A neural network approach using multi-scale textural metrics from very high-resolution panchromatic imagery for urban land-use classification," *Remote Sensing of Environment*, vol. 113, pp. 1276–1292, 2009, doi: 10.1016/j.rse.2009.02.014.

[28] I. A. Pradana, A. D. Rahajoe, and A. N. Sihananto, "Pengembangan Aplikasi Pendeteksi Keretakan Jalan Berbasis Android Dengan Implementasi Algoritma Hybrid CNN-LSTM," *Jurnal Informatika dan sistem Informasi (JIFOSI)*, vol. 5, no. 2, pp. 1–10, 2024, doi: https://doi.org/10.33005/jifosi.v5i2.146.

[29] M. Reyad and A. M. Sarhan, "A modified Adam algorithm for deep neural network optimization," *Neural Computing and Applications*, 2023, doi: 10.1007/s00521-023-08568-z.

Vol. 6, No. 3, Juni 2025, Page. 1217-1228
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.3.4435