Convolutional Neural Network based Hurricane Damage Detection using Satellite Images

Swapandeep Kaur
Chitkara University
Sheifali Gupta
Chitkara University
Swati Singh
Chitkara University
Deepika Koundal (koundal@gmail.com)
UPES: University of Petroleum and Energy Studies https://orcid.org/0000-0003-1688-8772
Atef Zaguia
Taif University

Research Article

Keywords: natural disaster, damage, hurricane, remote sensing, computer vision, convolutional neural network

Posted Date: September 29th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-934531/v1

Version of Record: A version of this preprint was published at Soft Computing on February 21st, 2022.
See the published version at https://doi.org/10.1007/s00500-022-06805-6.

Abstract

Huge swirling storms known as hurricanes are tropical storms appearing in the North Atlantic Ocean and Northeast Pacific that result in winds of 120 km/hour and higher. The winds occurring during hurricanes are catastrophic resulting in immense damage to human life and property. Rapid assessment of damage caused by hurricanes is extremely important for the first responders. But this process is usually slow, expensive, labor intensive and prone to errors. The advancements in remote sensing and computer vision help in observing Earth at a different scale. In this paper, a Convolutional Neural Network model has been designed that assesses the damage caused to buildings of post hurricane satellite images. The images have been classified as Damaged and Undamaged. The model is composed of five convolutional layers, five pooling layers, one flattening layer, one dropout layer and two dense layers. Hurricane Harvey dataset consisting of 23000 images of size 128 X 128 pixels has been used in this paper. The proposed model performed best at learning rate of 0.00001 and 30 epochs with the Adam optimizer obtaining an accuracy of 0.95, precision of 0.97, recall of 0.96 and F1-score of 0.96. It also achieved the best accuracy and minimum loss.

1. Introduction

An increase in the occurrence of natural disasters has been evident from the year 1980. People residing in the disaster-prone areas are also increasing. This leads to a hike in losses and damage caused due to the natural disasters [1].

A low-pressure region that develops over the tropical or subtropical waters is known as a tropical cyclone. In the Atlantic basin, the tropical cyclones are known as hurricanes. The energy of the hurricanes is drawn through the warm surface waters. When the warm and moist air spirals in counter clockwise direction inwards towards the center, there is an increase in the wind speed which reaches its maximum in the area surrounding the calm center of the hurricane [2]. The hurricanes are very fatal causing excessive damage to human life and property.

Hurricanes are storms that are cataclysmic in nature. Table 1 presents the storm's classification based on the wind speed. The storm is known as tropical storm when the wind speed is between 64 km/hour to 118 km/hour. When the wind speed is greater than 118 km/hour, it is known as a hurricane. A tropical depression occurs when the wind speed is below 64 km/hour. It becomes impossible to prevent an extreme weather event when the speed of the wind increases the above threshold.

Table 1
Classification of Storms Based on Wind Speed in Tropical Region

Type of Storms	Wind Speed (km/hour)
Tropical Storm	64–118
Hurricane	> 118
Tropical Depression	< 64

Satellite images are gaining popularity for monitoring of hurricanes [3]. Satellite images help in assessing the situation by providing an aerial view. But this process is still dependent on inspection by humans and is thus slow and unreliable. Hence, computer vision comes into the picture.

Recently, due to the revolutionizing of computing capacity, there has been tremendous development in deep learning (DL) [4]. DL with Convolutional Neural Network (CNN) performs extremely well in case of classification of images. The CNN model extracts feature from the images, therefore there is no requirement for any other feature extraction methods [5].

CNN is a kind of feedforward network in which the convolution operation is applied instead of the general matrix multiplication. It consists of mainly three layers: convolutional layer, pooling layer and fully connected layer [6]. The convolutional layer helps in extracting features from the image. Convolutional filters along with mathematical operations are applied to the input image which produces feature maps. ReLU (Rectified Linear Unit) is applied after the convolutional operation for introducing nonlinearities in the network. ReLU $f(x) = \max(0, x)$ helps in speeding up the training of the network without any effect on the network performance. Pooling or subsampling layers helps in reduction of the dimensions of the feature maps in order to decrease the processing time [7]. After the extraction of features and the reduction of resolution by the convolutional and pooling layers, the network is flattened into a feature vector by passing through the fully connected layers. Output obtained from convolution and pooling layers exhibit high level features which are used for classification by the fully connected layers [7].

The major contribution of this research paper is as following:

1. A CNN based model has been proposed that comprises five convolutional layers, five pooling layers, one flattening layer, one dropout layer and two dense layers.
2. The impact of various hyperparameters such as optimizers and learning rate has been studied. Finally, the best model is found that will be beneficial for finding out the damage caused due to hurricanes automatically.

The remaining portion of the paper is organized as related work in Sect. 2, proposed methodology in Sect. 3, results and discussion in Sect. 4, comparison with state-of-the-art models in Sect. 5 and conclusion in Sect. 6.

2. Related Work

Recently, DL has proved to be beneficial for automatic detection of damages caused by hurricanes.

The author introduced and evaluated CNN models for the damage caused by natural disasters from the aerial imagery. The CNN models were trained on aerial videos named Volan 2018. Eight CNN models were trained by transfer learning that achieved 80.69% mAP and 74.48 % for high altitude and low altitude respectively [1]. A benchmark dataset was created from the data available publicly. The dataset was created for the Greater Houston area of Hurricane Harvey that occurred in the year 2017. The dataset could be utilized by the researchers for training and test of the object detection models for automatic detection of the damaged buildings [8]. Satellite images along with deep learning are being used for various applications such as in disaster response and monitoring of the environment. CNNs with post processing neural networks combined CNN predictions along with satellite metadata. An accuracy of 0.83 and F1 score of 0.797 was obtained for the IARPA fMoW dataset [5]. A model for change detection using CNN was designed for finding out the areas affected severely by Hurricane Harvey and an F1 score of 81.2% was obtained. The satellite images were thresholded and clustered to make grids from which the disaster impact was determined [9]. Deep Phurie- a deep CNN model was proposed for automatic estimation of hurricane intensity of satellite images. A smaller root mean squared error (RMSE) of 8.82 knots was obtained as compared to 11.2 knots RMSE obtained from the previously built model – Phurie [2]. A semantic segmentation NN known as U -Net along with the ResNet model was used to determine the damage caused to roads for the Digital Globe satellite images of Hurricane Harvey. An accuracy of 0.845 and F1 -score 0f 0.675 was obtained [10]. VGG-16 CNN/ Multilayer perceptron was used to classify time period and urgency for Hurricane Harvey 2017 images [11]. The accuracy improved by 4% when various resolutions of CNN model were used for training it. Satellite images and CNN models were used for assessing the damage caused to buildings [12]. The author achieved an accuracy of 80.66 % and the root mean square of 10.18 knots were obtained when estimation of intensity was done of the HURDAT2 satellite dataset by using deep CNN [13]. mAP and mF1 of 44.83 and 56.025 were obtained when a single shot multibox detector method was used in determination of damage caused by Hurricane Sandy in the year 2012. For determination of damage from the post disaster images, a Convolutional Auto -Encoder consisting of VGG 16 was used [14]. An accuracy of 88.3% was achieved when a semi-supervised classification method was applied to the Hurricane Sandy dataset. This method consisted of three steps: segmentation, convolutional auto-encoder and fine-tuning using CNN. These techniques are used when there are more of unlabelled samples and less of labelled samples [15]. Various combinations of three neural networks were taken into account and it was found that when different colour masks of the relevant objects were taken, they performed in a better way. The first neural network was used for pre-processing and the second and third neural networks were useful for extracting features. These networks were used for detecting damage caused to buildings [16].

This paper is based on the choice of an optimal hyperparameter to reduce losses and improve performance of the proposed model.

3. Dataset Preparation

Hurricane Harvey dataset comprising 23000 satellite images has been used for the automatic damage detection. The images are further increased by the data augmentation techniques.

3.1 Dataset Analysis

Dataset used in this paper comprises the satellite images obtained after Hurricane Harvey that occurred in the Greater Houston region. The satellite images have been labeled as “damaged” and “undamaged” for the assessment of damage caused to buildings due to the hurricane. The images labelled as “damaged” indicate buildings affected by the hurricane and “undamaged” label indicate the buildings which were unaffected by the hurricane disaster. The number of “damaged “class images are 15000 and “undamaged” class images are 8000 in number. The dataset is divided into 15525 training images, 1725 validation images and 5750 testing images. Figure 1 shows damaged and undamaged sample images.

3.2 Dataset Pre-processing

Pre-processing is the most important stage in image processing because it causes an improvement in the features of the satellite images and also helps in suppressing unnecessary data present in the image [17] [18]. Normalization and data augmentation are the two steps under pre-processing.

3.2.1 Normalization

Data normalization is a very important step since it is used for maintaining numerical stability in the CNN models. Data normalization causes a CNN model to learn quickly and also makes its gradient descent stable. Hence, normalization of the pixel values of the images have been done in the range 0–1. This also makes the model unbiased in nature to the higher pixel values or feature values. The rescaling was done through multiplication of 1/255 along with the pixel values.

3.2.2 Data Augmentation

Data augmentation is used for expanding the size of the dataset [20]. The augmentation not only helps in increasing the dataset size but also is used for incorporating diversity in the dataset. This allows in improving the generalizability of the model. The model also becomes more robust when trained on slightly different new images [21][22].

Image Data Generator has been used for image augmentation. It augments the images in real time meaning that images are augmented on the fly during the training stage. This method saves a lot of memory. It returns only the images that have been transformed without adding to the original set of images. The images were increased from 23000 to 46800 images.

In this paper, data augmentation techniques such as zooming, rotation, horizontal flip and height/width shifting have been performed. In rotation technique, the image is rotated by an angle. Rearrangement of pixels takes place due to flipping while feature protection is kept intact.

The specifications of these data augmentation transformations are given in Table 2. The image is rotated by 40 degrees in the clockwise direction. 0.2 Width shift range is the upper bound of fraction of the total width range by which image will be either shifted right or left. Similarly, 0.2 height shift range specifies the fraction by which image will be shifted along the y-axis. A value of 0.2 zoom means the image will be zoomed between the range [0.8–1.2]. The image is flipped horizontally through the horizontal flip transformation.

Table 2
Data Augmentation Transformations

S.No.	Augmentation Method	Value
1	Rotation range	40
2	Width shift range	0.2
3	Height Shift range	0.2
4	Zoom range	0.2
5	Horizontal flip	True

3.3 Design of the Proposed CNN Model

A CNN model has been proposed for hurricane damage detection consisting of 15 layers. In our CNN model, consisting of five convolutional layers, five pooling layers, one dropout layer and two dense layers, there comes out to be 1,061,826 trainable parameters. The model depth is dependent on how complex are the features that are to be extracted out from the images. As our dataset consists of only two classes, a shallow network works well in terms of both generalization and the training time.

In the proposed CNN model, an input image is of 128*128*3 size that is applied to a convolutional layer of 32 filters generating an image of size 126*126*32. After that, max pooling layer of 2*2 pool size is applied generating an image of size 63*63*32. Further, convolutional layers comprising 64,128,128 and 256 filters are applied and each convolutional layer is followed by a max pooling layer.

After the max pooling layer, a flattening layer is applied. The flattening layer is followed by a dropout layer of 0.5. Dropout layer is used for making model distribution of the weights regular. The value of 0.5 in the dropout layer means that half or 50% of the neurons will be dropped randomly at each epoch. The addition of a dropout layer helps in reduction of overfitting of the CNN model [23]. The design of the CNN model has been shown in Fig. 3.

Table 3 shows the parameters in terms of filter size, image size and total number of parameters. An input image of 128*128*3 size is applied to a convolutional layer of size 128*128*3 of filter size 3*3 and comprising 32 filters. An output image of size 126*126*32 is obtained and 896 parameters are generated.

This image is then applied to a max pooling layer of pool size 2*2 which returns an output of size 63*63*32. This output is then given as input to the second convolutional layer comprising 64 filters that returns 18496 parameters. The output of size 61*61*64 is then given to the second max pooling layer that returns an output of size 30*30*64. The third convolutional layer returns an output of 28*28*128 size generating 73856 parameters. The third and the fourth convolutional layer comprised 128 filters. Each convolutional layer is followed by a max pooling layer. The fifth convolutional layer comprises 256 filters of size 3*3 and returned 295168 parameters. The fifth max pooling layer is followed by a flattening, dropout and two dense layers. The first dense layer returns 524800 parameters and the second dense layer returns 1026 parameters. The activation function of all the convolutional layers and the first dense layer is the ReLU function whereas the activation function of the last dense layer is the sigmoid function. ReLU activation function is faster, simpler and works well whereas sigmoid function helps to predict the probability of the output since its values lie between 0 and 1.

Table 3
Parameters of the Proposed CNN Model

	Layers	Input Image Size	Filter Size	No. of Filter	Activation Function	Output	Parameters
1	Input Image	128*128*3	–	–	–	–	–
2	Convolutional	128*128*3	3*3	32	ReLU	126*126*32	896
3	Max Pooling	126*126*32	Pool size (2*2)	–	–	63*63*32	0
4	Convolutional	63*63*32	3*3	64	ReLU	61*61*64	18496
5	Max Pooling	61*61*64	Pool size (2*2)	–	–	30*30*64	0
6	Convolutional	30*30*64	3*3	128	ReLU	28*28*128	73856
7	Max Pooling	28*28*128	Pool size (2*2)	–	–	14*14*128	0
8	Convolutional	14*14*128	3*3	128	ReLU	12*12*128	147584
9	Max Pooling	12*12*128	Pool size (2*2)	–	–	6*6*128	0
10	Convolutional	6*6*128	3*3	256	ReLU	4*4*256	295168
11	Max Pooling	4*4*256	Pool size (2*2)	–	–	2*2*256	0
12	Flatten	2*2*256	–	–	–	1024	0
13	Dropout	1024	–	–	–	1024	0
14	Dense	1024	–	–	ReLU	512	524800
15	Dense	512	–	–	sigmoid	2	1026

4. Results And Discussion

This section includes the discussion of results that were obtained from the proposed CNN model after change of the various hyperparameters like optimizers and learning rates. The learning rate is that hyperparameter that helps to control how much the model needs to change in response to the estimated error every time the weights of the model are updated. Choosing an appropriate value of learning rate is very important as a very small value can cause a longer training process and a very large learning rate

may result in an unstable training process. Optimizers are methods that help in minimizing the loss function or help in maximizing the efficiency. Optimizers are dependent on biases and weights. An optimizer helps in deciding how to change learning rates and weights so as to reduce the losses.

4.1 Performance Metrics

The various performance metrics [24–28] evaluated along with their equations are given below. TP stands for True Positive, FP stands for False Positives and TN and FN represent True Negative and False Negative respectively.

Accuracy – It is obtained by dividing the correct predictions by the total predictions as shown in Eq. 1.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} \quad (1)$$

Precision - This metric gives the predicted positive labels out of all the positive labels as shown in Eq. 2.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

Recall – Recall is obtained by dividing true predicted labels by the total estimated labels as shown in Eq. 3. It is also known as sensitivity.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

F1 – score – It is the harmonic mean of precision and recall as shown in Eq. 4.

$$\text{F1-score} = \frac{2XPrecisionXRecall}{Precision+Recall} \quad (4)$$

4.2 Specifications of Different Optimizers Used for Simulation

For classification of the hurricane images into damaged and undamaged classes, the proposed model is simulated using various deep learning optimizers. The optimizers used are Adagrad (Adaptive Gradient Algorithm), SGD (Stochastic Gradient Descent), RMSprop (Root Mean Squared Propagation) and Adam (Adaptive Moment Estimation). The learning specifications of the deep learning optimizers are given below in Table 4.

Table 4
Deep learning Optimizers' Learning Specifications

Optimizers	Specifications
SGD	learning rate = 1.E-03,, momentum = 0.9, decay = learning rate / EPOCHS
Adagrad	learning rate = 1.E-03, epsilon = 1e-07, decay = learning rate / EPOCHS
RMSprop	learning rate = 1.E-03, rho = 0.9, epsilon = 1e-07, decay = learning rate / EPOCHS
Adam	learning rate = 1.E-03, beta1 = 0.9,beta2 = 0.999, epsilon = 1e-07, decay = learning rate / EPOCHS

4.3 Results Analysis for Different Optimizers at Learning Rates 0.000001

This section describes the results of the experiment performed on the Hurricane Harvey dataset using a proposed CNN model with four different optimizers. The proposed model was analysed at learning rate = 0.000001 and 30 epochs.

4.3.1 Analysis of Accuracy and Loss

The performance in terms of training accuracy, training loss, training recall, validation accuracy, validation loss and validation recall for the various optimizers at learning rate of 0.000001 and 30 epochs is shown in Table 5. From the Table 5, it can be analysed that RMSProp performed best for training accuracy, training loss, training recall, validation accuracy and validation loss whereas Adam optimizer performed best for validation recall parameter.

Table 5
Analysis of Accuracy and Loss for Different Optimizers at LR = 0.000001

Optimizer	Training Accuracy	Training Loss	Training Recall	Validation Accuracy	Validation Loss	Validation Recall
Adagrad	0.5566	2.2566	0.6186	0.6464	1.0964	0.1977
SGD	0.5806	1.5373	0.5457	0.6510	0.7467	0.6678
RMSProp	**0.8234**	**0.4485**	**0.8150**	**0.8435**	**0.4076**	0.8446
Adam	0.7988	0.5057	0.7861	0.8058	0.4814	**0.8736**

Figure 4 shows the Model Accuracy for different Optimizers at each epoch with LR of 0.000001. Simulation is done for 30 epochs with four optimizers [29]. Figure 4 (a) shows the model accuracy for Adagrad [30] optimizer, Fig. 4 (b) shows for SGD optimizer [31], Fig. 4(c) shows model accuracy for

RMSProp optimizer [32] whereas Fig. 4 (d) shows model accuracy for Adam optimizer [33]. From the Fig. 4, it can be analysed that RMSProp performed best whereas Adagrad performed the worst in terms of model accuracy.

Figure 5 shows the Model Loss for different Optimizers at each epoch with LR of 0.000001. Simulation is done for 30 epochs with four optimizers. Figure 4 (a) shows the model loss for Adagrad optimizer, Fig. 4 (b) shows for SGD optimizer, Fig. 4(c) shows model loss for RMSProp optimizer whereas Fig. 4 (d) shows model loss for Adam optimizer. From the Fig. 5, it can be analyzed that RMSProp performed best whereas Adagrad performed the worst in terms of model loss.

4.3.2 Analysis of Different Confusion Matrix Parameters

Figure 6 shows the confusion matrix of the proposed model for different optimizers at a learning rate of 0.000001. Figure 6 (a),(b),(c),(d) shows the confusion matrices for the Adagrad, SGD, RMSProp and Adam optimizer respectively. In the confusion matrix, two classes are shown that are damaged and undamaged.

Table 6 shows the different confusion matrix parameters of the optimizers at a learning rate of 0.000001. From Table 6, it can be analysed that RMSProp performed best in terms of accuracy and F1-score, Adam performed best in terms of precision and Adagrad performed best in terms of recall.

Table 6
Confusion Matrix Parameters for various optimizers at LR = 0.000001

Optimizer	Accuracy (%)	Precision	Recall	F1-Score
Adagrad	0.64	0.65	**0.98**	0.78
SGD	0.66	0.68	0.91	0.78
RMSProp	**0.85**	0.87	0.91	**0.89**
Adam	0.83	**0.88**	0.86	0.87

The comparison of four optimizers in terms of confusion matrix parameters is shown graphically also in Fig. 7. From Fig. 7, it can be concluded that RMSProp and Adam are good in terms of accuracy, precision and F1-score whereas Adagrad performed best in terms of recall. Overall Rmsprop performed best and obtained an accuracy of 85%, precision of 87%, recall of 91% and F1-score of 89%.

4.4 Results Analysis for Different Optimizers at Learning Rates 0.00001

The performance in terms of training accuracy, training loss, training recall, validation accuracy, validation loss and validation recall for the various optimizers at learning rate of 0.00001 and 30 epochs is presented in this section.

4.4.1 Analysis of Accuracy and Loss

The analysis at a learning rate of 0.00001 and 30 epochs is shown in Table 7. From Table 7, it can be seen that Adam optimizer outperformed other optimizers as it achieved highest training and validation accuracy and recall and lowest loss. Adam optimizer performs the best as it is a combination of the best properties of both RMSProp and Adagrad optimizer.

Table 7
Training Performance of the Model for various optimizers at LR = 0.00001

Optimizer	Training Accuracy	Training Loss	Training Recall	Validation Accuracy	Validation Loss	Validation Recall
Adagrad	0.6556	0.8123	0.6176	0.7264	0.5749	0.7652
SGD	0.6757	0.6813	0.6546	0.7687	0.5859	0.7867
RMSProp	0.9484	0.1344	0.9477	0.9513	0.1213	0.9501
Adam	**0.9599**	**0.1076**	**0.9579**	**0.9519**	**0.1153**	**0.9519**

Figure 8 shows the Model Accuracy for different Optimizers at LR = 0.00001. The optimizers considered are Adagrad, SGD, RMSprop and Adam. Simulation is done for 30 epochs with four optimizers. Figure 8 (a) shows the model accuracy for Adagrad optimizer, Fig. 8(b) shows for SGD optimizer, Fig. 8(c) shows model accuracy for RMSProp optimizer whereas Fig. 8(d) shows model accuracy for Adam optimizer. From the Fig. 4, it can be analyzed that Adam optimizer performed best whereas Adagrad performed the worst in terms of model accuracy.

Figure 9 shows the Model Loss for different Optimizers at each epoch with LR of 0.00001. Simulation is done for 30 epochs with four optimizers. Figure 9 (a) shows the model loss for Adagrad optimizer, Fig. 9 (b) shows for SGD optimizer, Fig. 9(c) shows model loss for RMSProp optimizer whereas Fig. 9(d) shows model loss for Adam optimizer. From the Fig. 9, it can be analyzed that Adam performed best whereas Adagrad and SGD performed the worst in terms of model loss.

4.4.2 Analysis of Different Confusion Matrix Parameters

Figure 10 shows the confusion matrix of the proposed model for different optimizers at a learning rate of 0.000001. Figure 10 (a), (b), (c), (d) shows the confusion matrices for the Adagrad, SGD, RMSProp and Adam optimizer respectively.

In the confusion matrix, two classes are shown that are damaged and undamaged. Performance based on the different confusion matrix parameters such as accuracy, recall, precision and F1-score [34][35] has been discussed. Table 8 shows the classification performance of the four optimizers at learning rate = 0.00001 and 30 epochs. It was found that Adam outperformed other optimizers and achieved the best accuracy and precision of 0.95 and 0.97. An equal F1-score of 0.96 was obtained by both Adam and RMSProp optimizer. RMSProp optimizer achieved the best recall of 0.97.

Table 8
Confusion Matrix Parameters for different Optimizers at LR = 0.00001

Optimizer	Accuracy (%)	Precision	Recall	F1-Score
Adagrad	0.73	0.72	0.95	0.82
SGD	0.78	0.83	0.83	0.83
RMSProp	0.94	0.95	**0.97**	**0.96**
Adam	**0.95**	**0.97**	0.96	**0.96**

The comparison of four optimizers in terms of confusion matrix parameters is shown graphically also in Fig. 11.From Fig. 11, it can be concluded that RMSProp and Adam are good in terms of accuracy, precision, recall and F1-score. Overall Adam optimizer performed best and obtained an accuracy of 95%, precision of 97%, recall of 96% and F1-score of 96%.

4.5 Comparison of LR 0.000001 and LR 0.00001

Figure 12 shows the comparison of the confusion matrix parameters at learning rates 0.000001 and 0.00001. Figure 12 (a) shows comparison of accuracy, 12(b) comparison of precision, 12(c) comparison of recall and 12(d) shows the comparison of F1-score. It can be analysed from figure 12(a) that Adam optimizer achieves the best accuracy of 0.95 at learning rate of 0.00001. From figure 12 (b), it can be inferred that Adam optimizer obtains the best precision of 0.97 at learning rate of 0.00001. From figure 12 (c), it can be seen that Adagrad obtained the best recall of 0.98 at learning rate of 0.000001, followed by recall of 0.97 obtained by RMSProp and 0.96 obtained by Adam optimizer at learning rate of 0.00001. An equal F1-score of 0.96 is obtained by both RMSProp and Adam optimizer at a learning rate of 0.00001 as can be seen from figure 12(d). Thus, it can be inferred that the Adam optimizer is performing best at a learning rate of 0.00001.

5. Comparison Of The Proposed Model With State-of-the-art Models

The suggested model has been compared with the state-of-the-art models as shown in Table 9. The suggested model consisted of 15 layers and gave the best results at learning rate of 0.00001 and 30 epochs with the Adam optimizer.23000 images of damaged and undamaged classes were considered for the suggested model. The suggested model achieved an accuracy of 0.95, precision of 0.97, recall of 0.96 and F1-score of 0.96 which were better than the other models. Li et al. [14] worked on damage detection of 5041 images and obtained an F1-score of 56.025. Pradhan et al. [13] performed intensity estimation and obtained an accuracy of 80.66 %. Li et al. [15] obtained an accuracy of 88.3 % while finding out the damage caused due to Hurricane Sandy. Dotel et al. [3] obtained an accuracy of 84.5 % for damage detection of Hurricane Harvey. Doshi et al. [9] obtained a F1 score of 81.2% for Hurricane Harvey damage detection.

Table 9
Comparison of the Proposed Model with State of Art Models

Reference	Number of Classes	Number of Images	Image Size	Hurricane Name	Intensity Estimation/ Damage Detection	Results
Li et al. [14]	3	5041	1920x1080	Sandy	Damage Detection	F1-score – 56.025
Pradhan et al. [13]	5	48828	232x232	cyclones	Intensity Estimation	Accuracy-80.66%
Li et al. [15]	3	700	1920x1080	Sandy	Damage Detection	Accuracy-88.3%
Dotel et al. [3]	2	18474	9351x9351	Harvey	Damage Detection	Accuracy – 84.5 %
Doshi et al. [9]	2	1000	256x256	Harvey	Damage Detection	F1-score-81.2%
Proposed Model	2	23000	128x128	Harvey	Damage Detection	Accuracy-95% F1-score – 96%

6. Conclusion

In this paper, a convolutional neural network has been proposed for the automatic detection of damage caused to buildings using the satellite images due to the Hurricane Harvey that occurred in the Greater Houston region in the year 2017. Damage detection after the natural disasters is of prime importance for the first responders so that the people affected by the hurricane disaster can be provided aid at the earliest. The proposed model was made up of 15 layers consisting of five convolutional layers, five max pooling layers, one flattening layer, one dropout layer and two dense layers. Four optimizers that is Adagrad, SGD, RMSProp and Adam were compared at different learning rates and at 30 epochs. It was

found that the proposed model worked best with the Adam optimizer at learning rate of 0.00001 and 30 epochs. An accuracy of 0.95, precision of 0.97, recall of 0.96 and F1-score of 0.96 were obtained.

Declarations

Ethical approval: Ethical approval is not applicable to manuscript.

Funding: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

Informed Consent: Informed Consent is not applicable to manuscript.

Authorship Contributions

Swapandeep Kaur -idea for the article, performed the literature search, prepared the dataset, design the CNN model, conceived the experiment, wrote the Paper. **Sheifali Gupta and Swati Singh** – Conceptualization, drafted and critically revised the work, formal analysis, analyzing and interpretation of data, supervision. **Deepika Koundal, Atef Zaguia -** Formal analysis, drafted and critically revised the work, supervision.

References

[1] Pi, Y., Nath, N.D. and Behzadan, A.H., 2020. Convolutional neural networks for object detection in aerial imagery for disaster response and recovery. Advanced Engineering Informatics, 43, pp. 101009.

[2] Dawood, M. and Asif, A., 2019. Deep-PHURIE: deep learning based hurricane intensity estimation from infrared satellite imagery. Neural Computing and Applications, pp. 1-9.

[3] Dotel, S., Shrestha, A., Bhusal, A., Pathak, R., Shakya, A. and Panday, S.P., 2020, March. Disaster Assessment from Satellite Imagery by Analysing Topographical Features Using Deep Learning. In Proceedings of the 2020 2nd International Conference on Image, Video and Signal Processing (pp. 86-92).

[4] Kaur, S., Gupta, S., Singh, S. and Gupta, I., 2021. Detection of Alzheimer's Disease Using Deep Convolutional Neural Network. International Journal of Image and Graphics, pp. 2140012.

[5] Pritt, M. and Chern, G., 2017, October. Satellite image classification with deep learning. In 2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR) pp. 1-7. IEEE.

[6] Cao, Q.D. and Choe, Y., 2020. Building damage annotation on post-hurricane satellite imagery based on convolutional neural networks. Natural Hazards, 103(3), pp. 3357-3376.

[7] Phung, V.H. and Rhee, E.J., 2019. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. Applied Sciences, 9(21), p.4500

[8] Chen, S.A., Escay, A., Haberland, C., Schneider, T., Staneva, V. and Choe, Y., 2018. Benchmark dataset for automatic damaged building detection from post-hurricane remotely sensed imagery. arXiv preprint arXiv:1812.05581.

[9] Doshi, J., Basu, S. and Pang, G., 2018. From satellite imagery to disaster insights. arXiv preprint, arXiv:1812.07033.

[10] Dotel, S., Shrestha, A., Bhusal, A., Pathak, R., Shakya, A. and Panday, S.P., 2020, March. Disaster Assessment from Satellite Imagery by Analysing Topographical Features Using Deep Learning. In Proceedings of the 2020 2nd International Conference on Image, Video and Signal Processing, pp. 86-92.

[11] Robertson, B.W., Johnson, M., Murthy, D., Smith, W.R. and Stephens, K.K., 2019. Using a combination of human insights and 'deep learning'for real-time disaster communication. Progress in Disaster Science, 2, p.100030.

[12] Duarte, D., Nex, F., Kerle, N. and Vosselman, G., 2018. Satellite Image Classification Of Building Damages Using Airborne And Satellite Image Samples In A Deep Learning Approach. ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences, 4(2).

[13] Pradhan, R., Aygun, R.S., Maskey, M., Ramachandran, R. and Cecil, D.J., 2017. Tropical cyclone intensity estimation using a deep convolutional neural network. IEEE Transactions on Image Processing, 27(2), pp.692-702.

[14] Li, Y., Hu, W., Dong, H. and Zhang, X., 2019. Building damage detection from post-event aerial imagery using single shot multibox detector. Applied Sciences, 9(6), p.1128.

[15] Li, Y., Ye, S. and Bartoli, I., 2018. Semisupervised classification of hurricane damage from postevent aerial imagery using deep learning. Journal of Applied Remote Sensing, 12(4), pp.045008.

[16] Nia, K.R. and Mori, G., 2017, May. Building damage assessment using deep learning and ground-level image data. In 2017 14th conference on computer and robot vision (CRV) (pp. 95-102). IEEE.

[17] Scannell, C.M., Veta, M., Villa, A.D., Sammut, E.C., Lee, J., Breeuwer, M. and Chiribiri, A., 2020. Deep-learning-based preprocessing for quantitative myocardial perfusion MRI. Journal of Magnetic Resonance Imaging, 51(6), pp.1689-1696.

[18] Zheng, X., Wang, M. and Ordieres-Meré, J., 2018. Comparison of data preprocessing approaches for applying deep learning to human activity recognition in the context of industry 4.0. Sensors, 18(7), p.2146.

[19] Ba, J.L., Kiros, J.R. and Hinton, G.E., 2016. Layer normalization. arXiv preprint arXiv:1607.06450.

[20] Perez, L. and Wang, J., 2017. The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621.

[21] El Naqa, I., Li, R. and Murphy, M.J. eds., 2015. Machine learning in radiation oncology: theory and applications. Springer.

[22] Shin, H.C., Roth, H.R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D. and Summers, R.M., 2016. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE transactions on medical imaging, 35(5), pp.1285-1298.

[23] Li, Z., Gong, B. and Yang, T., 2016. Improved dropout for shallow and deep learning. Advances in neural information processing systems, 29, pp.2523-2531.

[24] Denil, M., Shakibi, B., Dinh, L., Ranzato, M.A. and De Freitas, N., 2013. Predicting parameters in deep learning. arXiv preprint arXiv:1306.0543.

[25] Betz, J.M., Brown, P.N. and Roman, M.C., 2011. Accuracy, precision, and reliability of chemical measurements in natural products research. Fitoterapia, 82(1), pp.44-52.

[26] Ng, B., Quinete, N. and Gardinali, P.R., 2020. Assessing accuracy, precision and selectivity using quality controls for non-targeted analysis. Science of The Total Environment, 713, p.136568.

[27] Handelman, G.S., Kok, H.K., Chandra, R.V., Razavi, A.H., Huang, S., Brooks, M., Lee,M.J. and Asadi, H., 2019. Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods. American Journal of Roentgenology, 212(1), pp.38-43.

[28] Hossin, M. and Sulaiman, M.N., 2015. A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & Knowledge Management Process, 5(2), p.1.

[29] Choi, D., Shallue, C.J., Nado, Z., Lee, J., Maddison, C.J. and Dahl, G.E., 2019. On empirical comparisons of optimizers for deep learning. arXiv preprint arXiv:1910.05446.

[30] Lydia, A. and Francis, S., 2019. Adagrad—an optimizer for stochastic gradient descent. Int. J. Inf. Comput. Sci, 6(5).

[31] Duda, J., 2019. SGD momentum optimizer with step estimation by online parabola model. arXiv preprint arXiv:1907.07063.

[32] Wichrowska, O., Maheswaranathan, N., Hoffman, M.W., Colmenarejo, S.G., Denil, M., Freitas, N. and Sohl-Dickstein, J., 2017, July. Learned optimizers that scale and generalize. In International Conference on Machine Learning (pp. 3751-3760). PMLR.

[33] Kumar, A., Sarkar, S. and Pradhan, C., 2020. Malaria disease detection using cnn technique with sgd, rmsprop and adam optimizers. In Deep Learning Techniques for Biomedical and Health Informatics (pp. 211-230). Springer, Cham.

[34] Fürnkranz, J. and Flach, P.A., 2003. An analysis of rule evaluation metrics. In Proceedings of the 20th international conference on machine learning (ICML-03) (pp. 202-209).

[35] Zhou, J., Gandomi, A.H., Chen, F. and Holzinger, A., 2021. Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics, 10(5), p.593.

Figures

Figure 1

Sample Hurricane images (a) Damaged (b) Undamaged

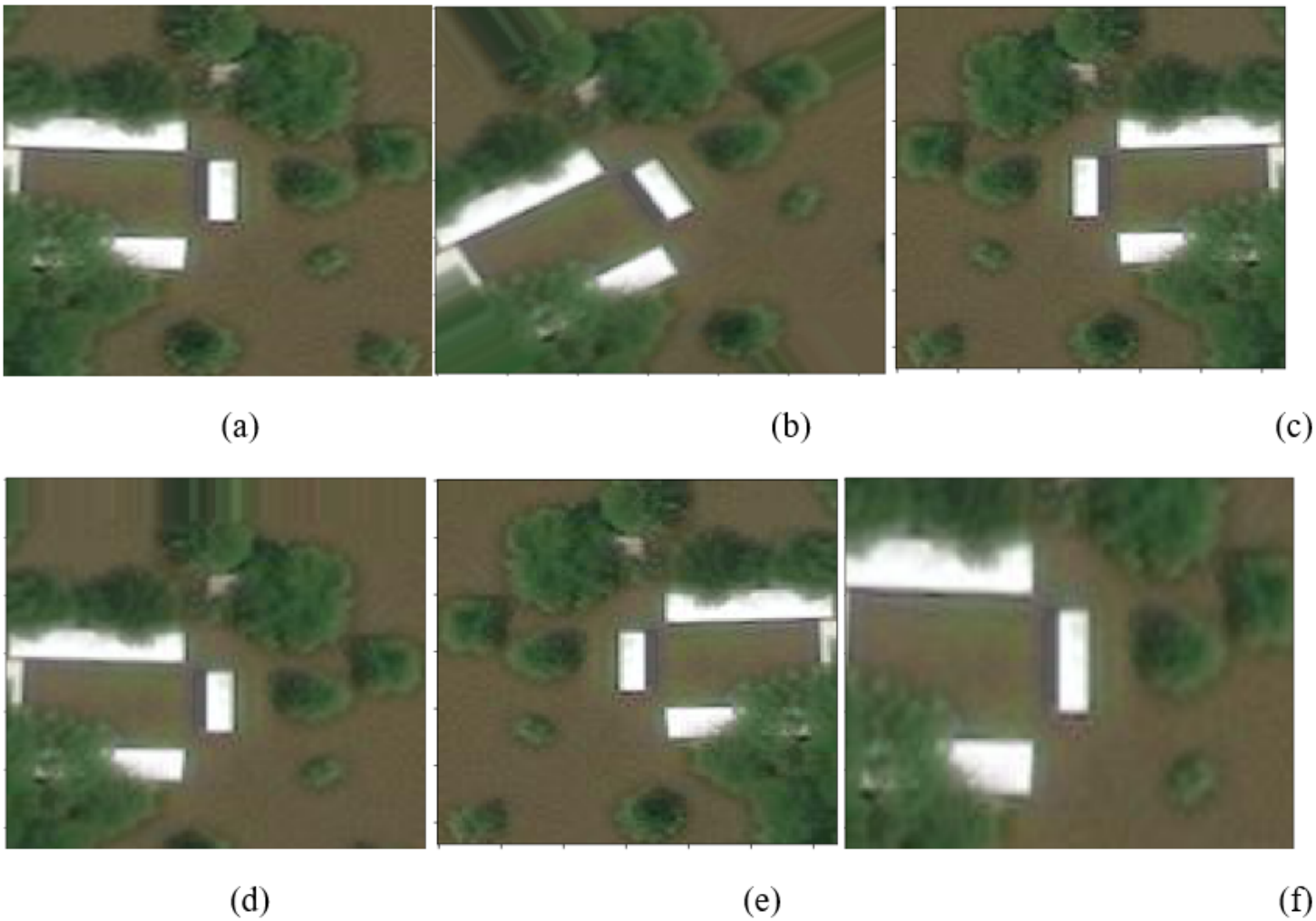

Figure 2

Sample results of data augmentation of Input Image (a) Original Image (b) Rotated Image (c) Width Shifting (d) Height Shifting (e) Horizontal Flipping (f) Zoomed Image

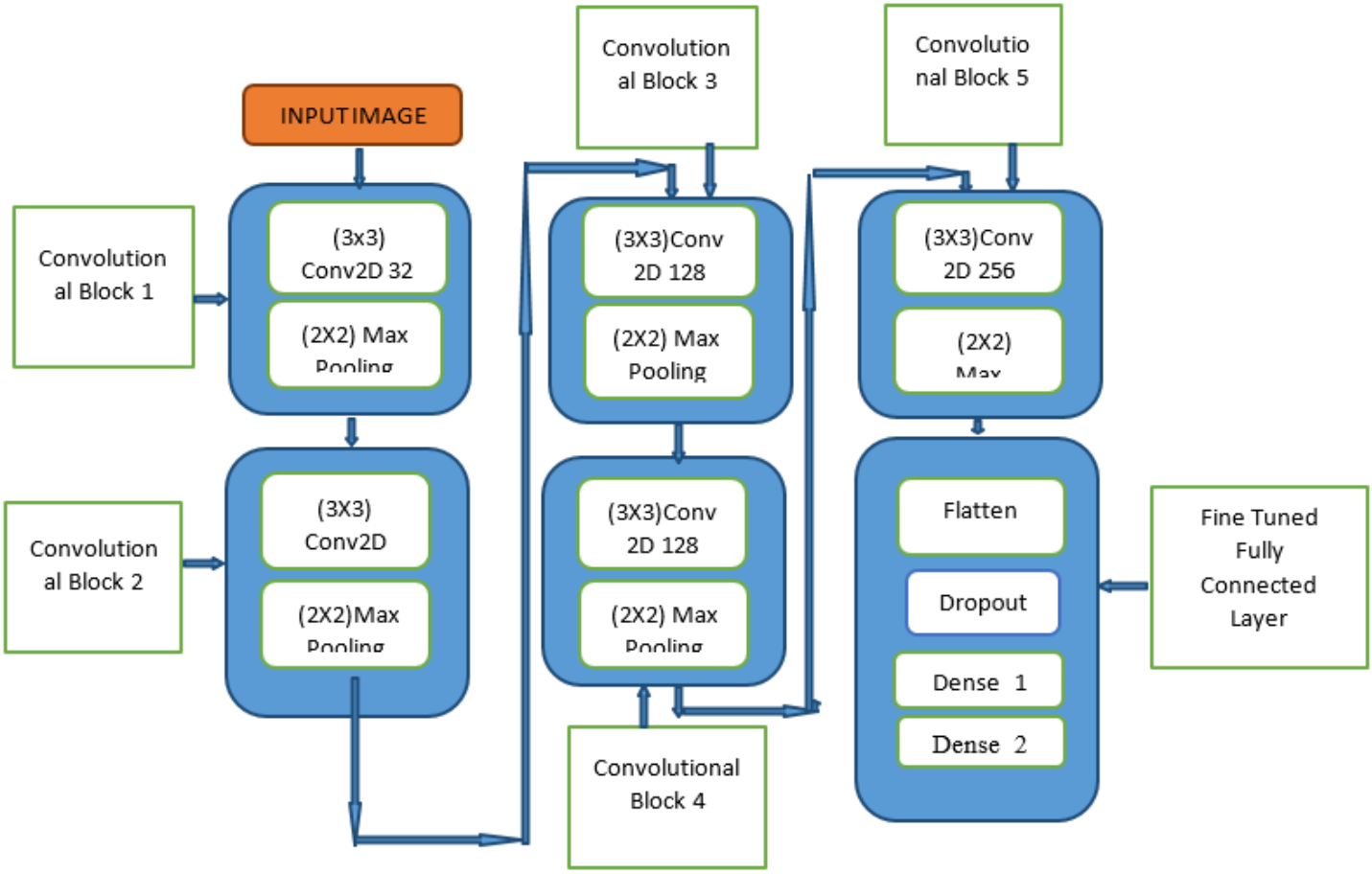

Figure 3

Design of the Proposed CNN Model

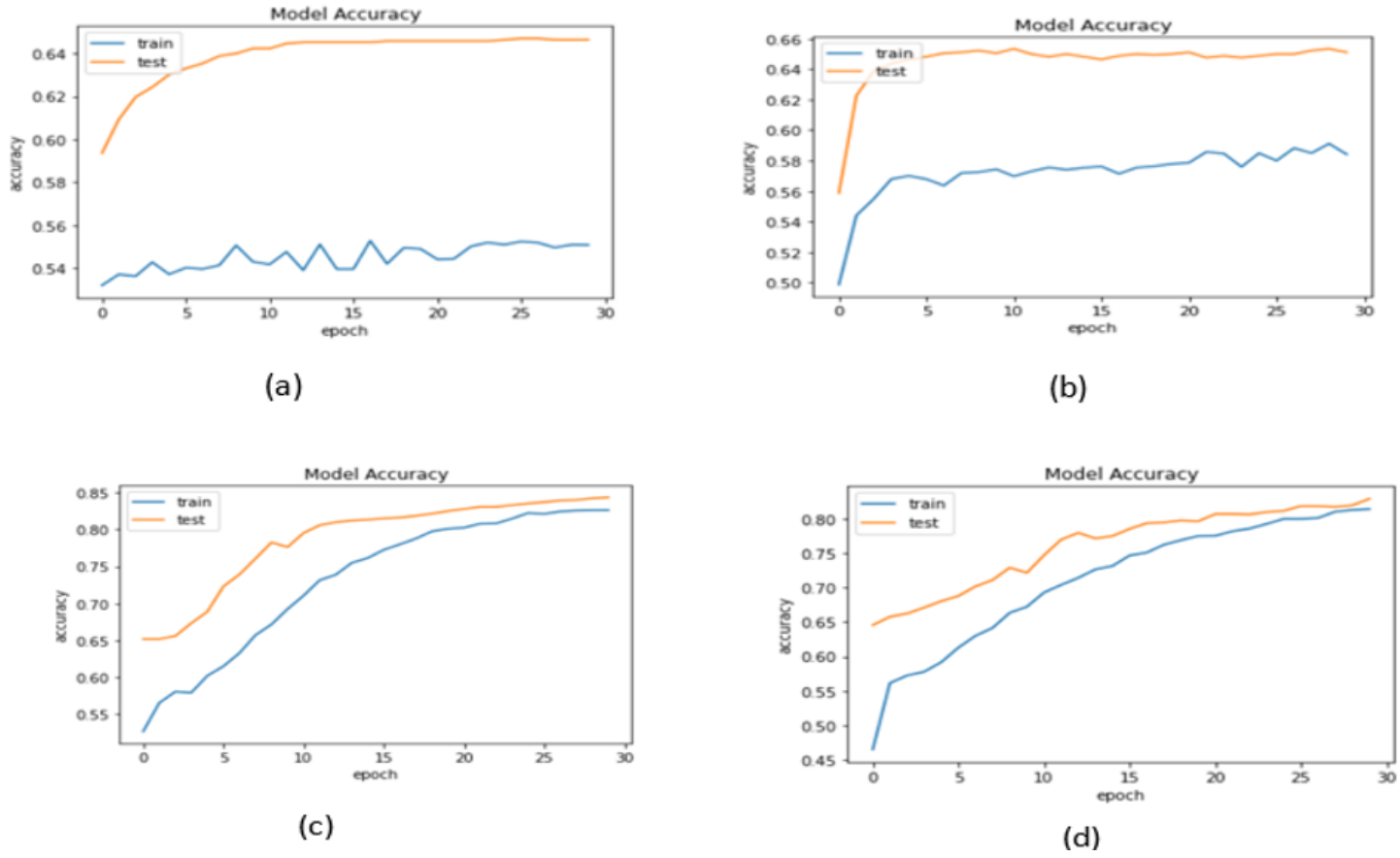

Figure 4

Model Accuracy for Different Optimizers at LR= 0.000001 (a) Adagrad (b) SGD (c) RMSProp (d)Adam

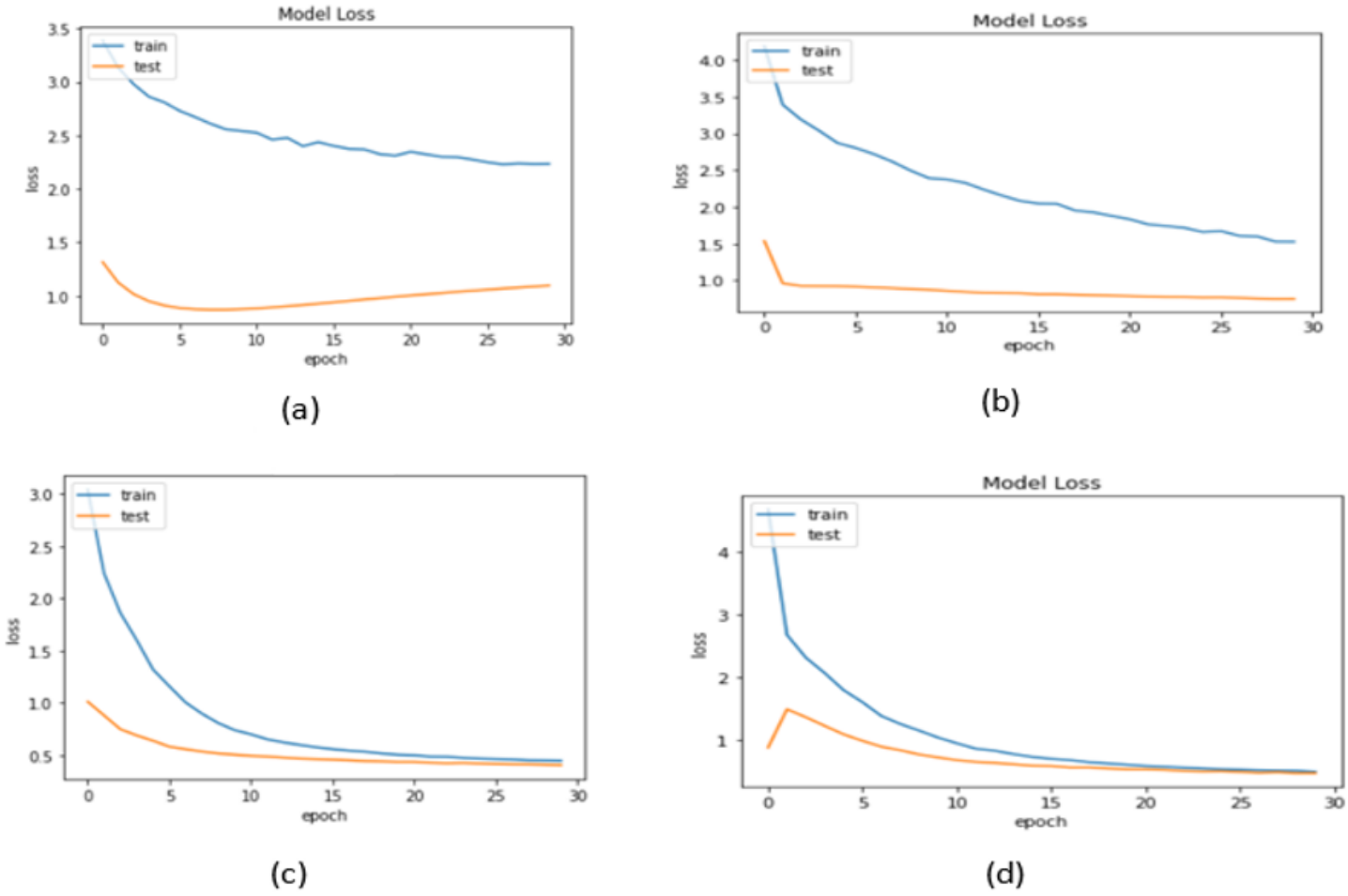

Figure 5

Model Loss for Different Optimizers at LR= 0.000001 (a) Adagrad (b) SGD (c) RMSProp (d) Adam

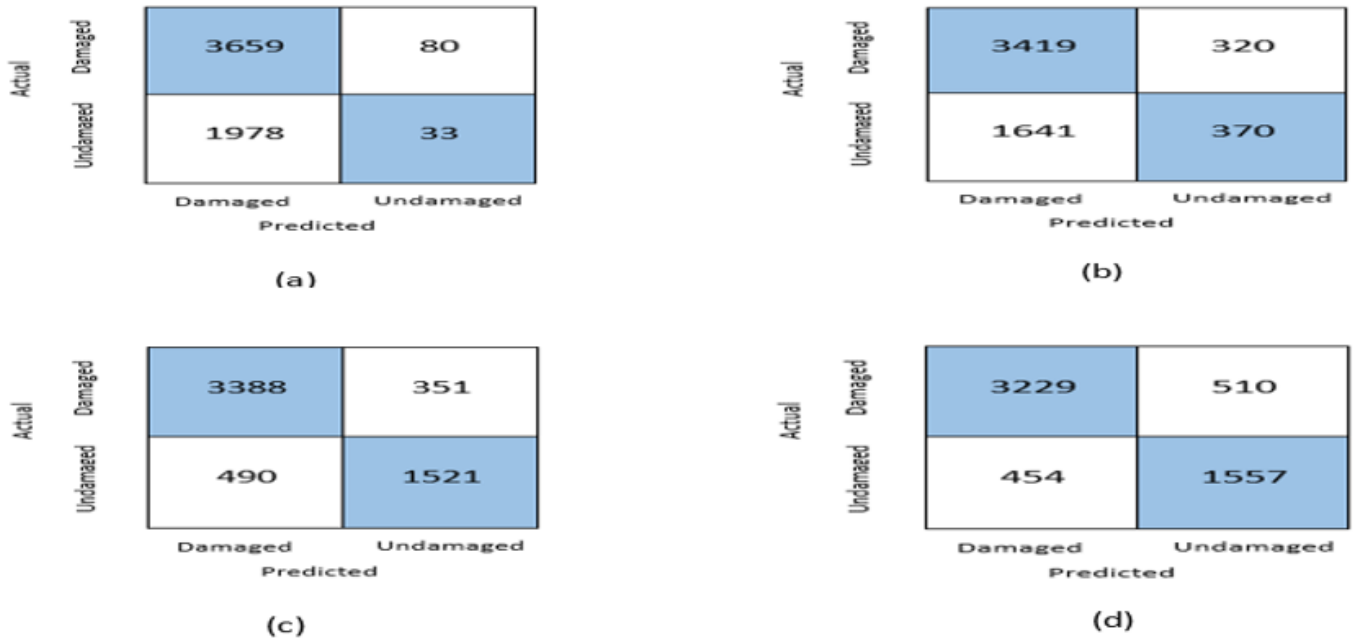

Figure 6

Confusion Matrix for Different Optimizers at LR= 0.000001 (a) Adagrad (b) SGD (c) RMSProp (d) Adam

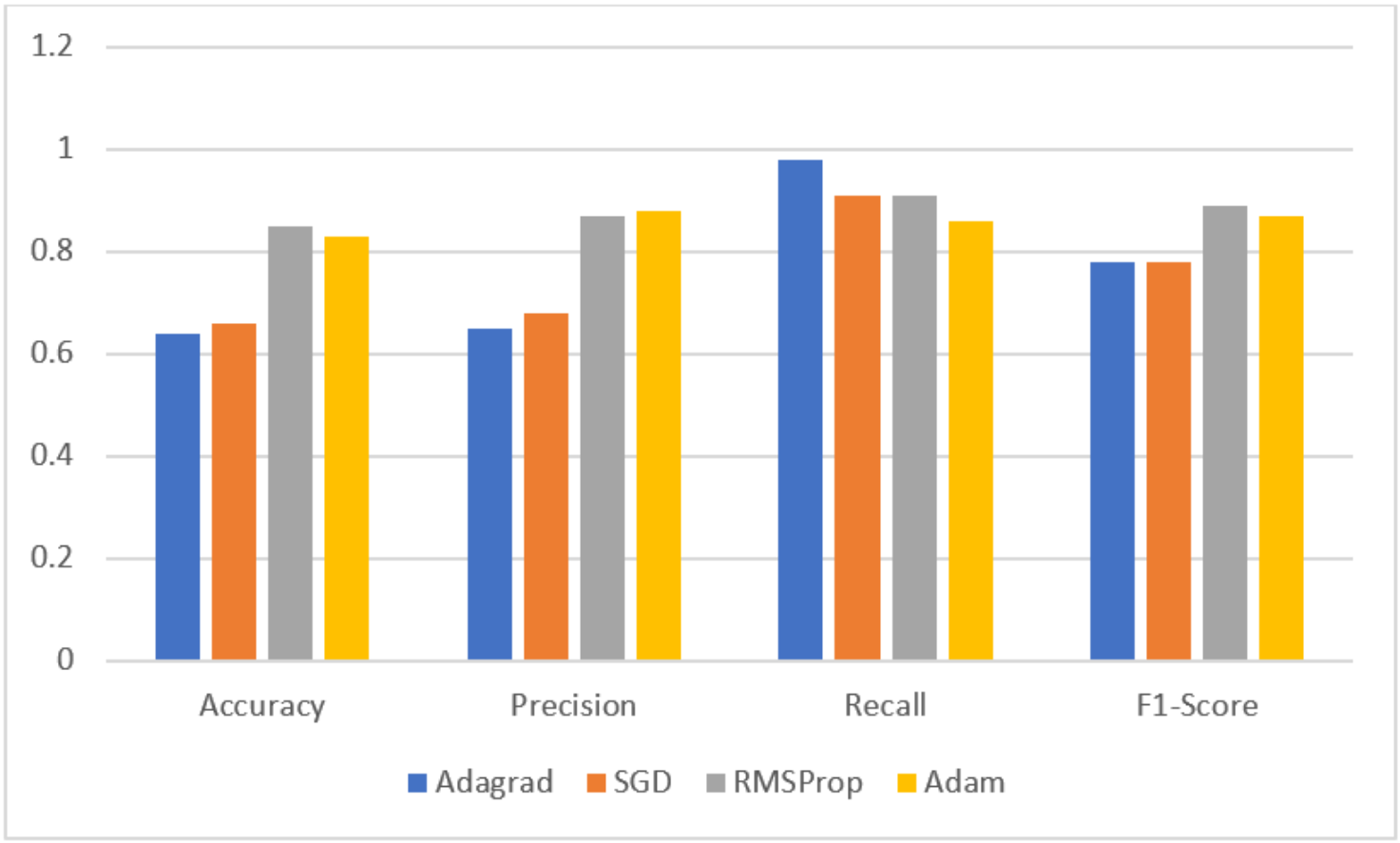

Figure 7

Analysis of Different Confusion Matrix Parameters at LR= 0.000001

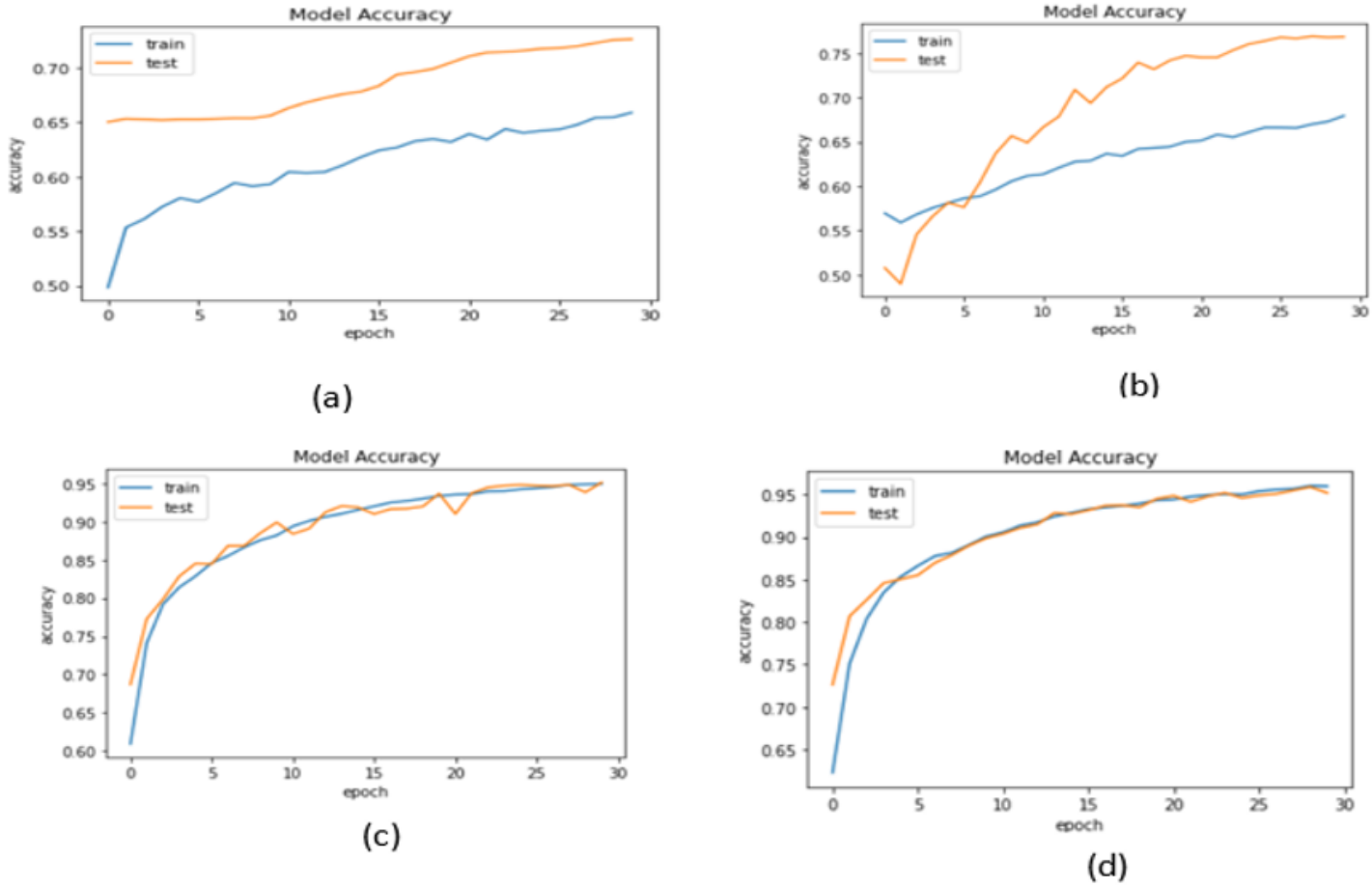

Figure 8

Model Accuracy for Various Optimizers at LR= 0.00001 and 30 Epochs (a) Adagrad (b) SGD (c) RMSProp (d) Adam

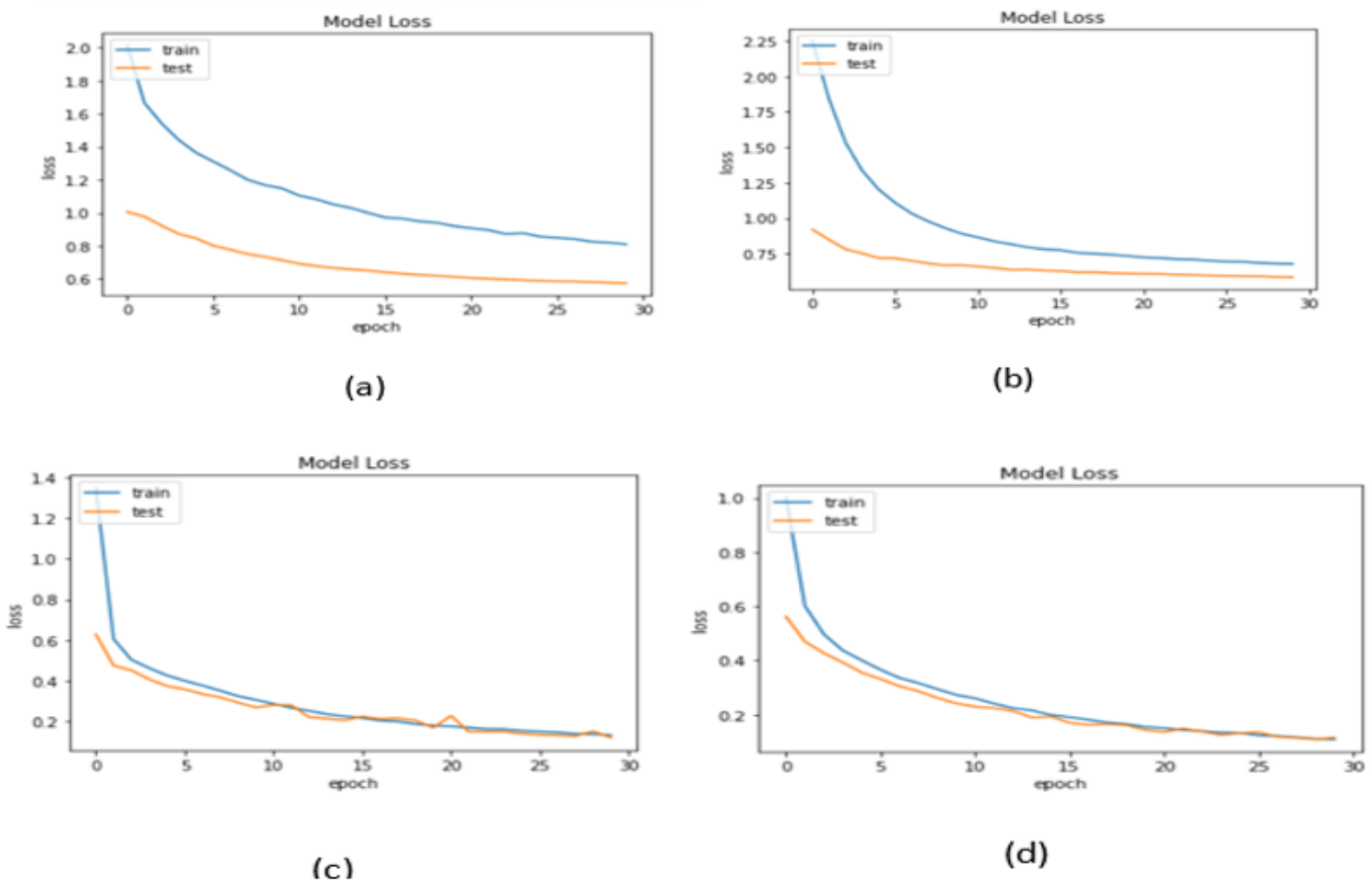

Figure 9

Model Loss for Different Optimizers at LR= 0.00001 and 30 Epochs (a) Adagrad (b) SGD (c) RMSProp (d) Adam

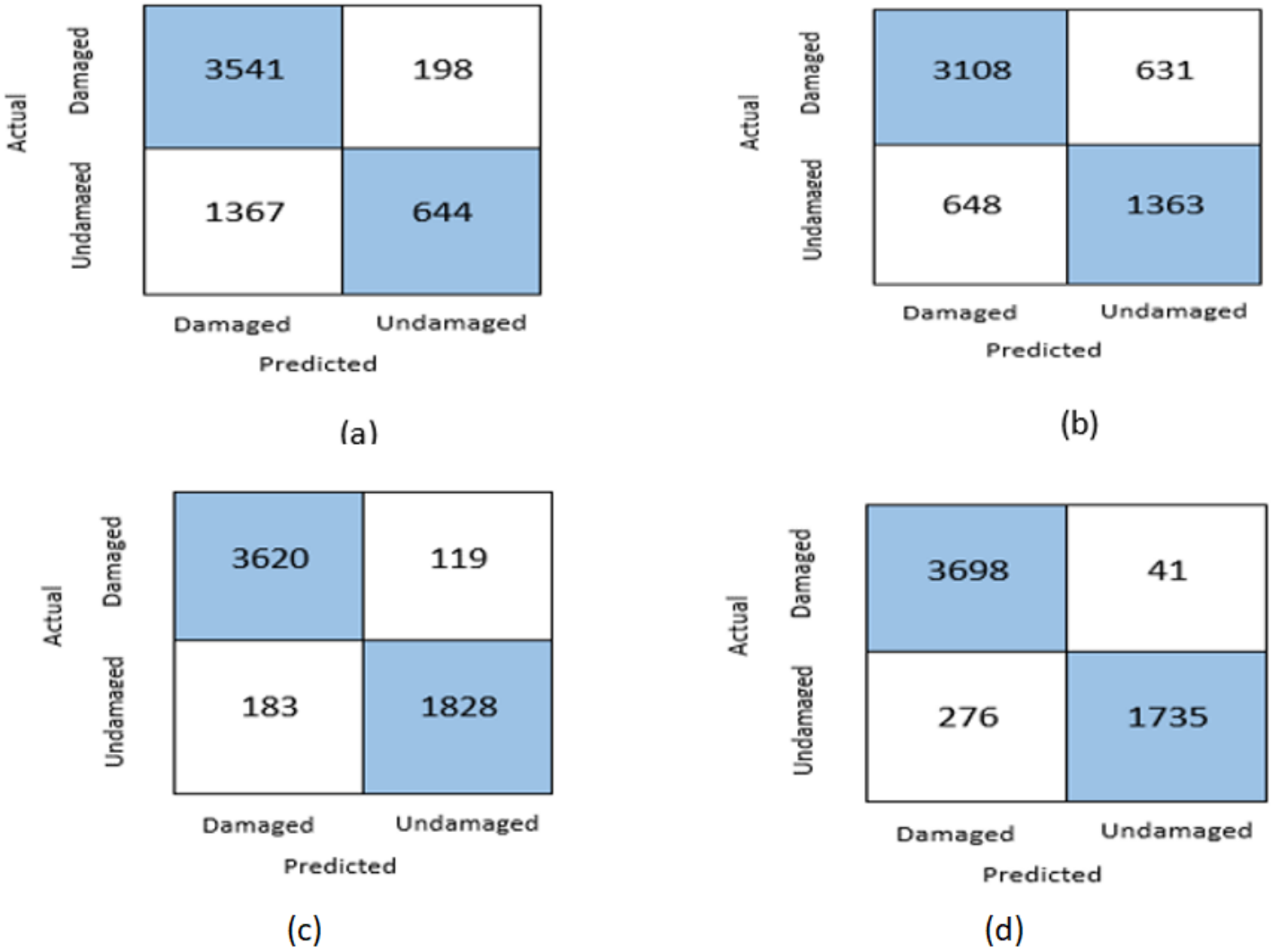

Figure 10

Confusion Matrix for Various Optimizers at LR= 0.00001 (a) Adagrad (b) SGD (c) RMSProp (d) Adam

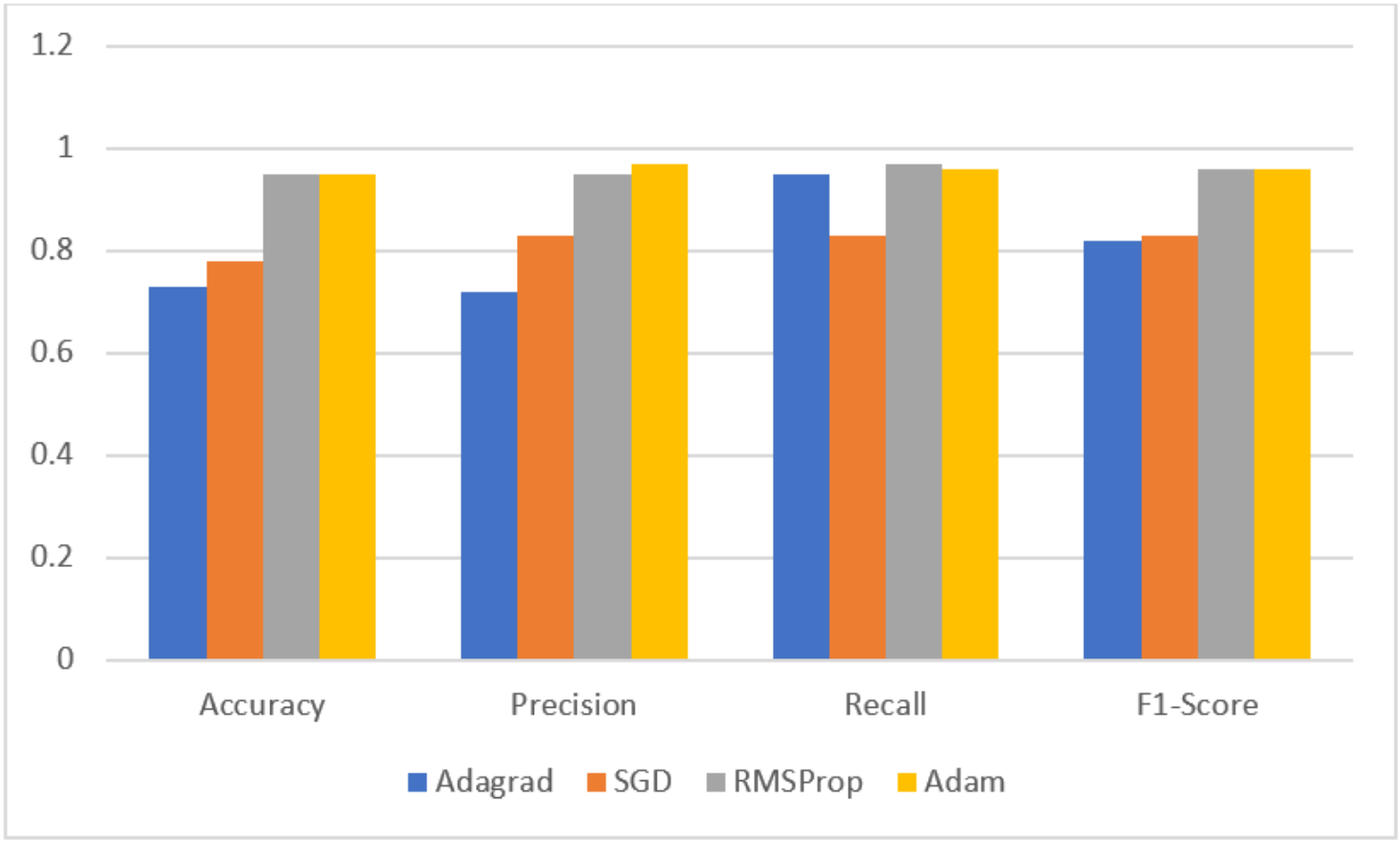

Figure 11

Analysis of Different Confusion Matrix Parameters at LR= 0.00001

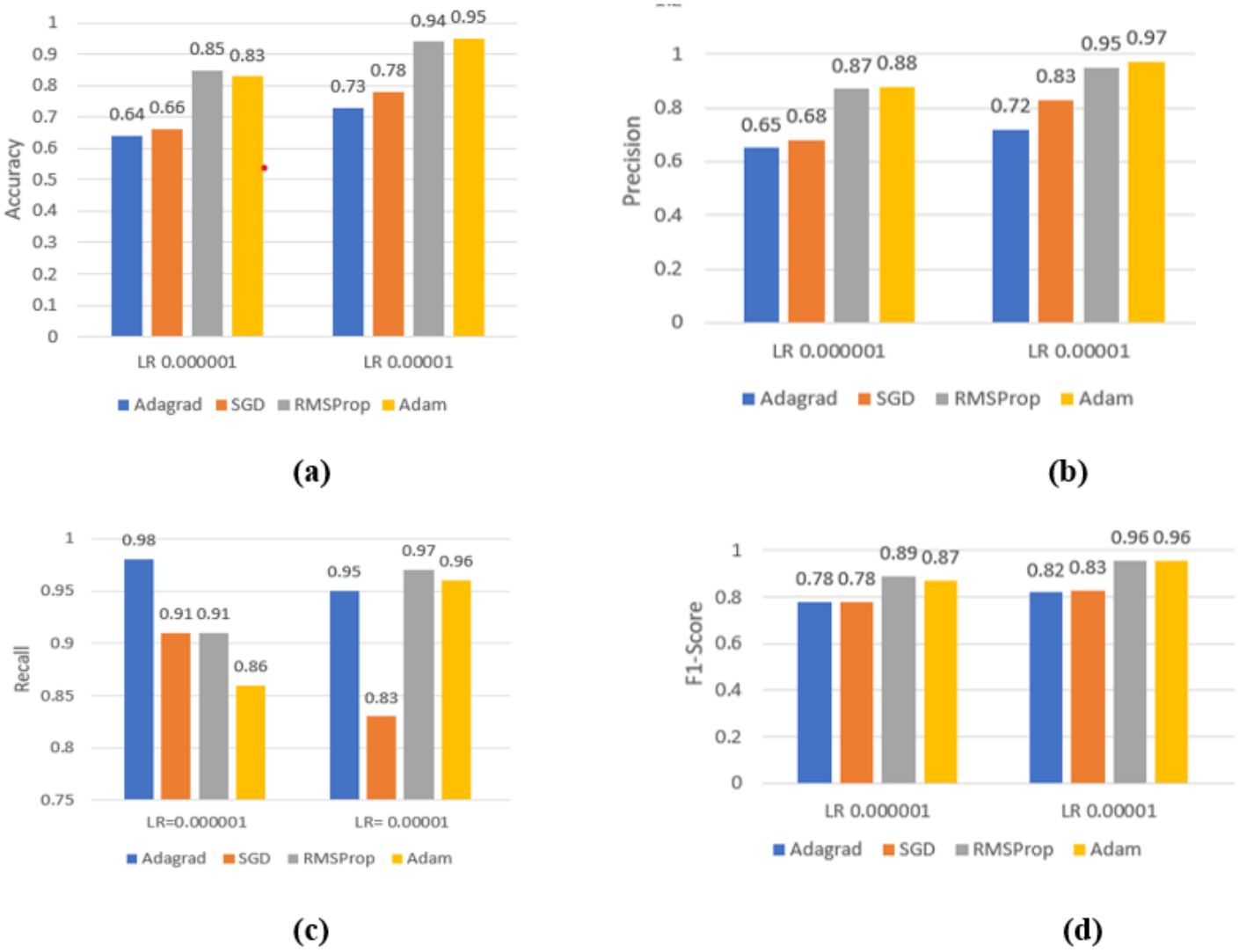

Figure 12

Comparison of Confusion Matrix Parameters at Learning Rate of 0.000001 and 0.00001 (a) Accuracy (b) Precision (c) Recall (d) F1-Score