IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 December 2023, accepted 27 January 2024, date of publication 1 February 2024, date of current version 8 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3361287

RESEARCH ARTICLE

Building Damage Assessment Using Feature Concatenated Siamese Neural Network

MGS M. LUTHFI RAMADHAN[1], GRAFIKA JATI[1,2], AND WISNU JATMIKO[1], (Senior Member, IEEE)
[1]Faculty of Computer Science, University of Indonesia, Depok 16424, Indonesia
[2]Department of Electrical, Electronic, and Information Engineering "Guglielmo Marconi" (DEI), University of Bologna, 40126 Bologna, Italy

Corresponding author: Mgs M. Luthfi Ramadhan (mgs.m01@ui.ac.id)

This work was supported by Publikasi Terindeks Internasional (PUTI) Q1 from Universitas Indonesia for research project entitled "Asesmen Kerusakan Bangunan Akibat Bencana Gempa Bumi Menggunakan Residual Siamese Neural Network Pada Data Lidar" under Grant NKB-395/UN2.RST/HKP.05.00/2022.

ABSTRACT Fast and accurate post-earthquake building damage assessment is an important task to do to define search and rescue procedures. Many approaches have been proposed to automate this process by using artificial intelligence, some of which use handcrafted features that are considered inefficient. This research proposed end-to-end building damage assessment based on a Siamese neural network. We modify the network by adding a feature concatenation mechanism to enrich the data feature. This concatenation mechanism creates different features based on each output from the convolution block. It concatenates them into a high-dimensional vector so that the feature representation is more likely to be linearly separable, resulting in better discrimination capability than the standard siamese. Our model was evaluated through three experimental scenarios where we performed classification of G1 or G5, G1-G4 or G5, and all the five grades of EMS-98 building damage description. Our models are superior to the standard Siamese neural network and state-of-the-art in this field. Our model obtains f1-scores of 79.47%, 54.09%, 40.64% and accuracy scores of 87.24%, 95.28%, and 42.57% for the first, second, and third experiments, respectively.

INDEX TERMS Classification, deep learning, disaster, earthquake, LiDAR, siamese neural network.

I. INTRODUCTION

Buildings are an essential aspect of the life of every human being. Most people live and work in buildings. However, buildings are very vulnerable to natural disasters. An earthquake, for example, is known to very often cause damage to buildings and even take lives [1]. Most of the injuries and casualties are caused by falling debris from the building [2], [3]. Therefore, it is important for the rescue team to immediately observe the condition of the building after the earthquake to assist the disaster response process and rescue operations [4], [5].

Previously, the post-earthquake observation process was carried out by conducting field surveys, which were time-consuming, costly, and terrible for creating building damage maps and planning rescue operations. Moreover, in some cases, this is not possible to do due to broken or blocked roads caused by the earthquake making some locations impossible to be accessed [1], [6], [7], [8]. Especially, for extensively devastated areas, getting the results right away after the disaster is no longer feasible [9].

Alternatively, Remote sensing and geographical information system (GIS) technology can be applied to accelerate this work as it can capture affected areas from space [10], [11]. However, doing manual observations using remote sensing and GIS technology still takes a lot of time if the area affected by the disaster is wide [12]. Therefore, the post-earthquake building observation process needs to be automated.

Previous researches [1], [13] utilize artificial intelligence based on satellite imagery to automate post-earthquake building observation. However, there are certain types of collapsed patterns that cannot be captured using images, one of which is pancake collapse where the building collapsed vertically with an intact rooftop [14]. light detection and ranging (LiDAR) point cloud data can be used to address this problem since it has height information.

Previous researches that based on LiDAR data [5], [6] use support vector machine (SVM) with a handcrafted feature

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

that is not robust and requires human expertise. Therefore this research utilizes a Siamese neural network that has a convolutional neural network (CNN) as its sub-network to automatically perform feature extraction. Fujita et al. [9] use an unshared weight siamese neural network (SNN) to detect tsunami-whased buildings because the appearance of the pre-tsunami and the post-tsunami image is dissimilar. However, in the case of an earthquake, the appearance of the pre-earthquake and post-earthquake building is not as dissimilar as tsunami-washed, since the building is either damaged or collapsed instead of washed away. This research decided to keep the weight sharing since the idea is to perform classification based on the change after the disaster occurred. Moreover, the weight sharing makes our model very analogous to previous researches that use LiDAR on earthquake data.

As of what has been found by [5] that the difference between pre-earthquake feature and post-earthquake feature is the most informative feature, therefore we apply some modification to the SNN to enrich this feature. We perform three experimental scenarios, one of which is a five-class classification using all five Grades of EMS-98 building damage description to further detail the post-earthquake observation. As far as the author is concerned, this research is the only one that classifies building damage in the Kumamoto Prefecture earthquake that used all five Grades of EMS-98 based on LiDAR data.

The main contributions of this paper lie in the following three aspects:

1) We introduced a novel architecture based on an SNN by implementing a concatenation mechanism to enrich the difference features by concatenating each feature map from the low level all the way to the high level.
2) The proposed architecture outperforms its baseline and other models in most metrics.
3) We performed five-class classification using all five Grades of EMS-98 building damage description on LiDAR data.

The remainder of this paper is organized as follows. Section two gives a brief overview of related work. Section three gives a brief overview of artificial neural networks (ANN). The dataset and its description are detailed in Section Four. Section five details our proposed method. The result and discussion are provided in Section Six. We finally conclude this paper in Section Seven.

II. RELATED WORK

A. IMAGE-BASED BUILDING DAMAGE ASSESSMENT

Several researches have been conducted to automate post-disaster building observations by utilizing artificial intelligence technology. Ji et al. [1] utilizes pre-earthquake and post-earthquake images to detect collapsed buildings after an earthquake disaster in Port-au-prince, Haiti 2010. The pre-earthquake and post-earthquake images are concatenated depth-wise into a tensor which is then performed feature extraction. Min Ji compares two feature extraction techniques, namely CNN and Gray Level Cooccurrence Matrix (GLCM) which are then classified using random forest. The experimental results prove that CNN feature extraction has greater accuracy than GLCM feature extraction with an accuracy of 87% and 85% for CNN and GLCM respectively. A similar research was conducted by Miura et al. [13], this time it was carried out to automate the assessment of post-earthquake damage to buildings in the Kumamoto Prefecture 2016 and Kobe 1995 areas based on the post-earthquake image only. The classification was carried out using CNN and obtained an accuracy score of 93%.

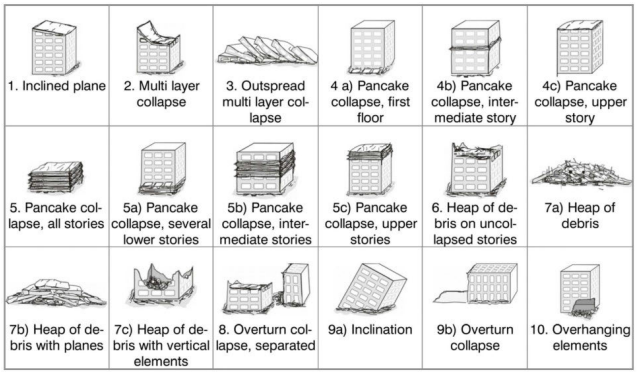


FIGURE 1. Compilation of collapsed building types [14].

Both Ji et al. [1] and Miura et al. [13] use image data in assessing building damage, but the drawback of using images lies in not detecting buildings that collapse vertically with a perfect roof (pancake collapsed) as illustrated in part 4b in Fig. 1 [14]. The same thing was also stated by Min Ji that it requires LiDAR data to capture the height reduction pattern of the building after the earthquake occurred. By using LiDAR data, pancake collapse can be detected because of the representation of the height in the LiDAR data. Therefore, in this research, the authors focused on LiDAR data in assessing building damage.

B. LIDAR POINT CLOUD-BASED BUILDING DAMAGE ASSESSMENT

Several researches have been utilizing LiDAR data to assess building damage. Hajeb et al. [6] conducted an assessment of building damage using pre-earthquake LiDAR data and post-earthquake LiDAR data in Kumamoto Prefecture. Both the pre-earthquake and post-earthquake data are transformed into the digital surface model (DSM). The DSM projected the point cloud to its z-axis resulting in a matrix where each element of this matrix is the z-axis of each point in the point cloud. Then, the GLCM feature extraction was applied both to the pre-earthquake DSM and post-earthquake DSM. After that, the resulting GLCM feature was subtracted between pre-earthquake and post-earthquake by using element-wise subtraction, based on these differences the classification was done by using a SVM and achieved an accuracy score of 87%. A similar research was conducted by Moya et al. [5]. Using

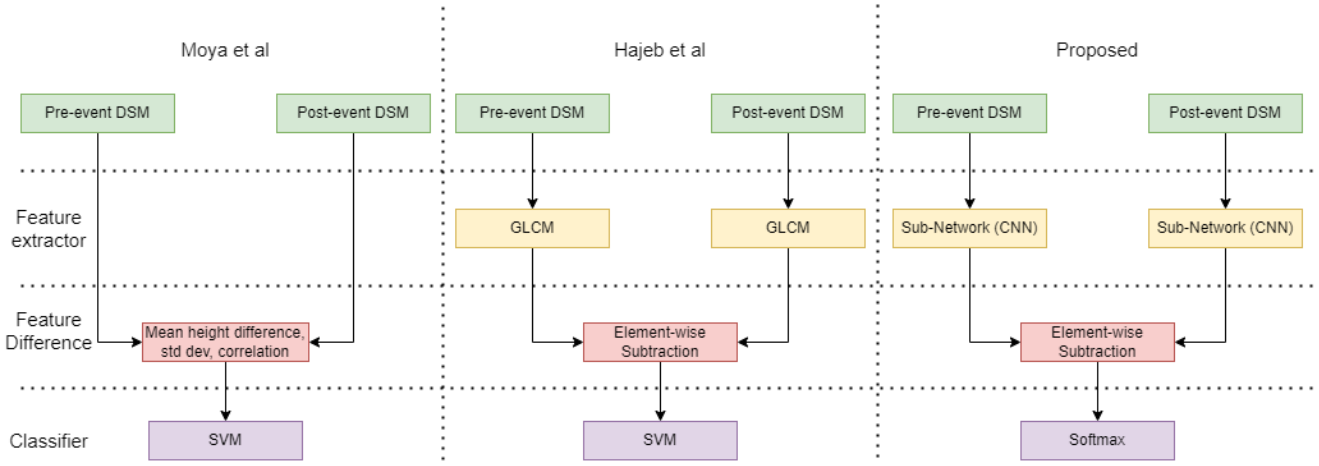


FIGURE 2. Comparison of building damage assessment between previous methods and the proposed method.

903 buildings from the same disaster, they utilize mean height difference, standard deviation, and correlation coefficient as feature extraction techniques. The classification was done using SVM and achieved an accuracy score of 93%.

To simplify, what both Moya et al. [5] and Hajeb et al. [6] done essentially consists of three processes namely feature extraction, feature difference, and classification (see Fig. 2). First, the feature is extracted by using some feature extraction techniques, then the resulting feature between pre-earthquake and post-earthquake is subtracted using element-wise subtraction, and lastly, a classifier is trained to learn the pattern. These three processes are very analogous to an SNN. The feature extraction is done automatically by using a CNN as a sub-network of the SNN, on top of that an element-wise subtraction layer is stacked to calculate the difference between the feature vector of the pre-earthquake and post-earthquake, lastly, an output layer is stacked after element-wise subtraction layer to perform the classification task. The advantage of using an SNN is that it has CNN to automatically extract feature out of the data, therefore we don't need to perform manual feature extraction techniques which itself often raise a problem on its own about which feature extraction is the best fit for the given dataset since there are a lot of manual feature extraction techniques available [15], [16], [17]. Moreover, it has an end-to-end connection that allows the gradient from the loss function to reach every layer of the SNN. Therefore, the classification layer and feature extraction layer (CNN) can learn the pattern simultaneously.

In this research, the SNN is customized by implementing a feature concatenation mechanism which we explained further in section V-B. This customized SNN is the main contribution of this research, we named it a feature concatenated siamese neural network (FCSNN).

III. OVERVIEW OF ARTIFICIAL NEURAL NETWORK (ANN)

ANN is a subset of machine learning that takes inspiration from the human brain [18]. The human brain consists of billions of interconnected neurons that transmit electrical signals called neurotransmitters through connections to each other which leads to thought formation. This gives humans the ability to think and perform actions.

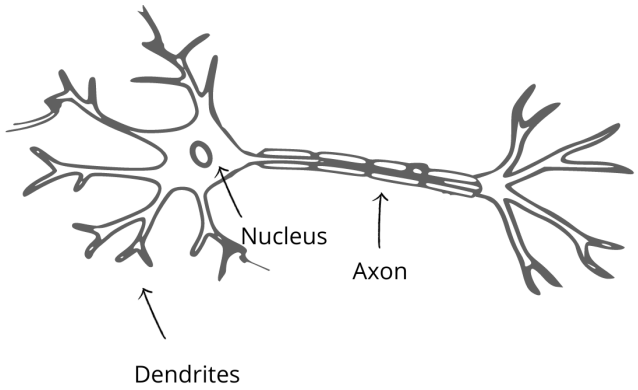


FIGURE 3. Biological neuron.

Each of these neurons consists of three primary parts namely dendrites, axons, and nucleus as illustrated in Fig. 3 [19]. Dendrites are the places where neurons receive input from other neurons. An axon is the output of a neuron that aims to transmit signals to other neurons. Meanwhile, the nucleus is where the signal is processed. Neurons communicate with each other by propagating the neurotransmitters across a narrow space called synapses located between the axon of the sending neuron and the dendrites of the receiving neuron.

A. ARTIFICIAL NEURON

The goal of ANN is to mimic the human brain with the hope of developing a machine that can behave and think like a human. Therefore the neuron in ANN composition is made identical to biological neuron. The dendrites are mimicked by the so-called weights and biases which compute the dot product between weights and incoming signal. The nucleus is mimicked by an activation function that maps the resulting

dot product to a certain range. Lastly, the Axon is mimicked by the output connection which propagates the resulting activation function to the next layer [20]. An artificial neuron is illustrated in Fig. 4.

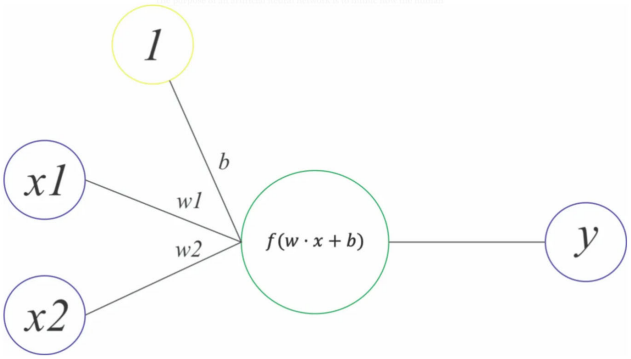


FIGURE 4. Artificial neuron.

Mathematically speaking, the operation in an artificial neuron can be formulated as in the equation (1).

$$y = f(\vec{x} \cdot \vec{w} + b) \tag{1}$$

where y represents the output of the artificial neuron, $\vec{w}$ represents the weights vector, b represents the bias, $\vec{x}$ represents the input vector, and f represents the activation function.

B. ARTIFICIAL NEURAL NETWORK (ANN)

ANN is a bunch of artificial neurons interconnected to each other. The neurons in ANN are structured into several sequential layers known as input, hidden, and output layers as illustrated in Fig. 5. Each neuron of a layer is connected to every neuron on the previous and the next layer, this is usually called a fully connected layer or dense layer. Every layer receives input from the previous layer, computes it, and propagates the result to the next layer. The input layer takes input and propagates it to the next layer. It doesn't compute anything therefore it doesn't have any weight, bias, and activation function. The hidden layer learns the complex pattern in the data and transforms them into a new representation that will ease the output layer to learn. The output layer provides the final output of the data that is being fed [21].

ANN learns by processing the input signals through the whole network and obtaining the result on the output layer. The output of ANN is then compared with the ground truth using a loss function. This loss function is a measurement that tells how well the ANN models the data. The higher the loss, the worse the model. It then nudges its weights and biases so that it has a better prediction. The most popular algorithm is gradient descent, which is an iterative process that aims to minimize the loss function. In other words, it tries to find weights and biases in which if those weights and biases are used, the loss function will be minimal.

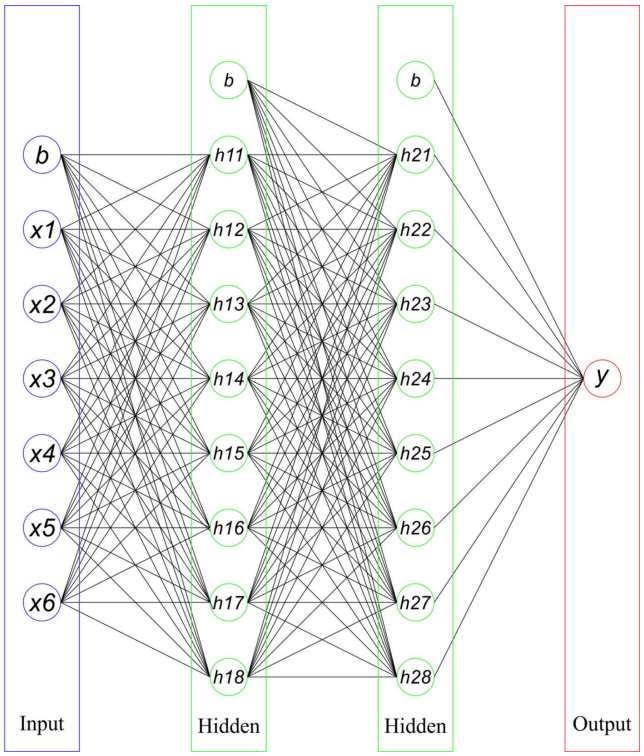


FIGURE 5. Illustration of an artificial neural network (ANN).

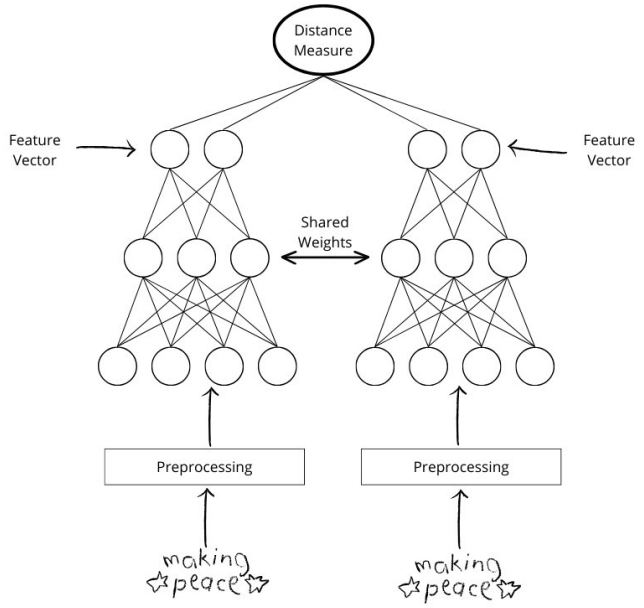


FIGURE 6. Siamese neural network architecture re-illustrated from [22].

C. SIAMESE NEURAL NETWORK (SNN)

SNN was first introduced by Bromley et al. [22] to solve biometric problems such as hand signature verification. This neural network architecture receives two inputs and produces an output in the form of the similarity of the two inputs. This architecture consists of two sub-networks that are identical to each other as illustrated in Fig. 6. The sub-network shares exactly the same architecture, weights, and biases.

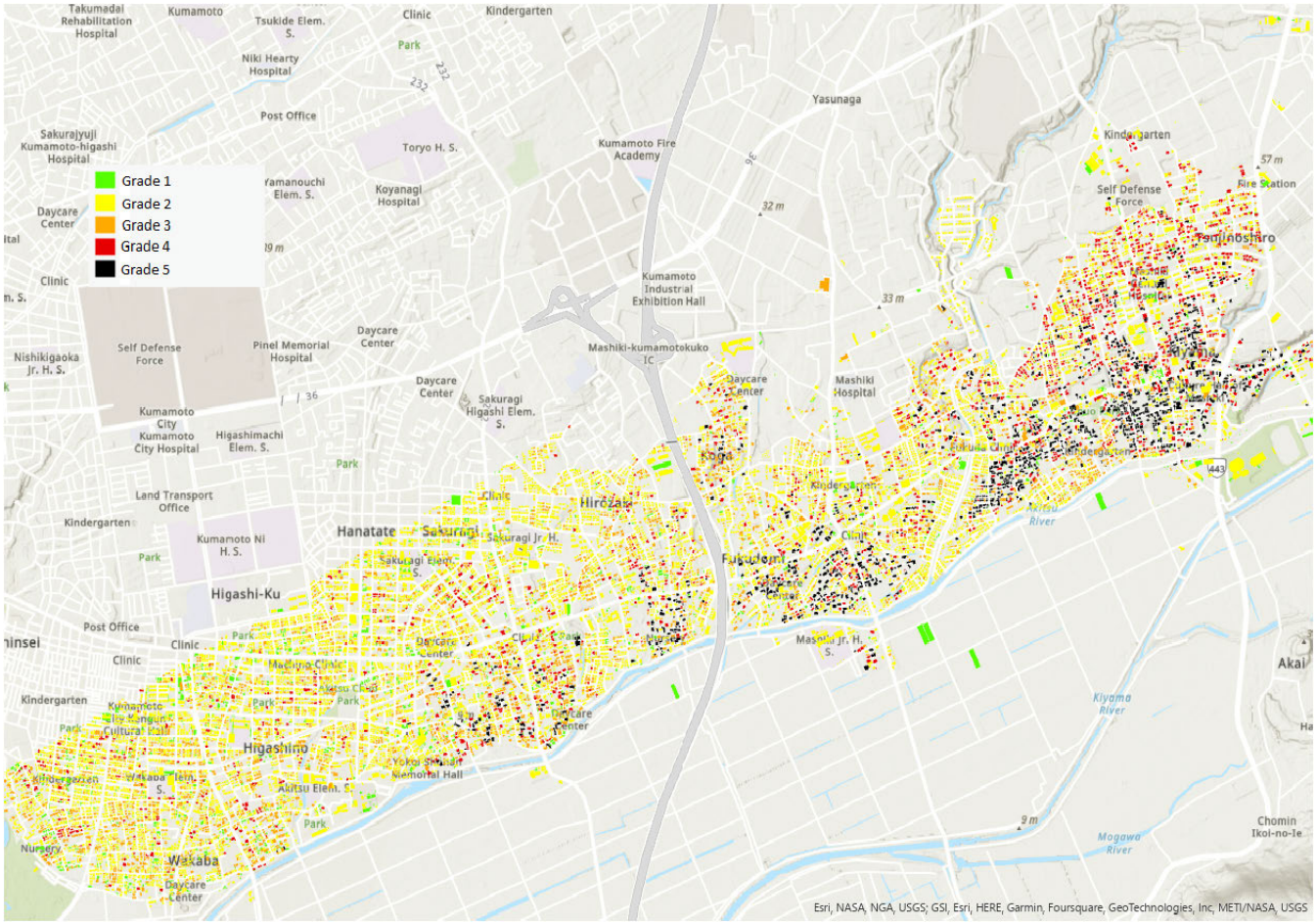


FIGURE 7. Distribution of building footprints and its labels.

This aims to ensure that the model can extract features with exactly the same techniques and computation for the two different inputs. The output of these two sub-networks is two feature vectors which are then fed to the distance measure layer (output layer) to calculate the similarity between the two vectors. Finally, in the case of biometric verification, thresholding is carried out to determine whether the query image is accepted or rejected. Although this model was originally invented for biometric verification, Many studies have adopted this model for problems in different fields ranging from medical images, robotics, natural language processing, and others. This model is also not limited to similarity measurement problems, but can also be applied to segmentation, classification, one-shot learning problems, and others [23]. For example, Mehmood et al. [24] used SNN to classify Alzheimer's disease. In the field of remote sensing, this model is very suitable for change detection tasks such as what has been done by [9] which classify buildings based on their change after disaster.

IV. DATASET

This research uses a LiDAR point cloud dataset downloaded from OpenTopography. This dataset is provided by Asia Air Survey Co., Ltd which contains point clouds in the Kumamoto Prefecture, Japan before and after the 2016 earthquake [25], [26]. The distribution of building footprints and their labels are illustrated in Fig. 7.

TABLE 1. The amount of buildings for each class.

Class	Amount	Percentage(%)
Grade 1	2.558	±14.36%
Grade 2	9.331	±52.40%
Grade 3	3.145	±17.66%
Grade 4	1.643	±9.22%
Grade 5	1.128	±6.33%

There are 17,805 building footprints with a total of five classes. Classes are the level of damage of buildings starting from G1 which is not damaged at all to G5 which is a complete collapse. These levels of damage are determined based on the EMS-98 building damage description illustrated in Fig. 8 while the amount of buildings for each class is listed in Table. 1.

In this research, the dataset is divided into training, testing, and validation data with a ratio of 60:20:20 using a stratified

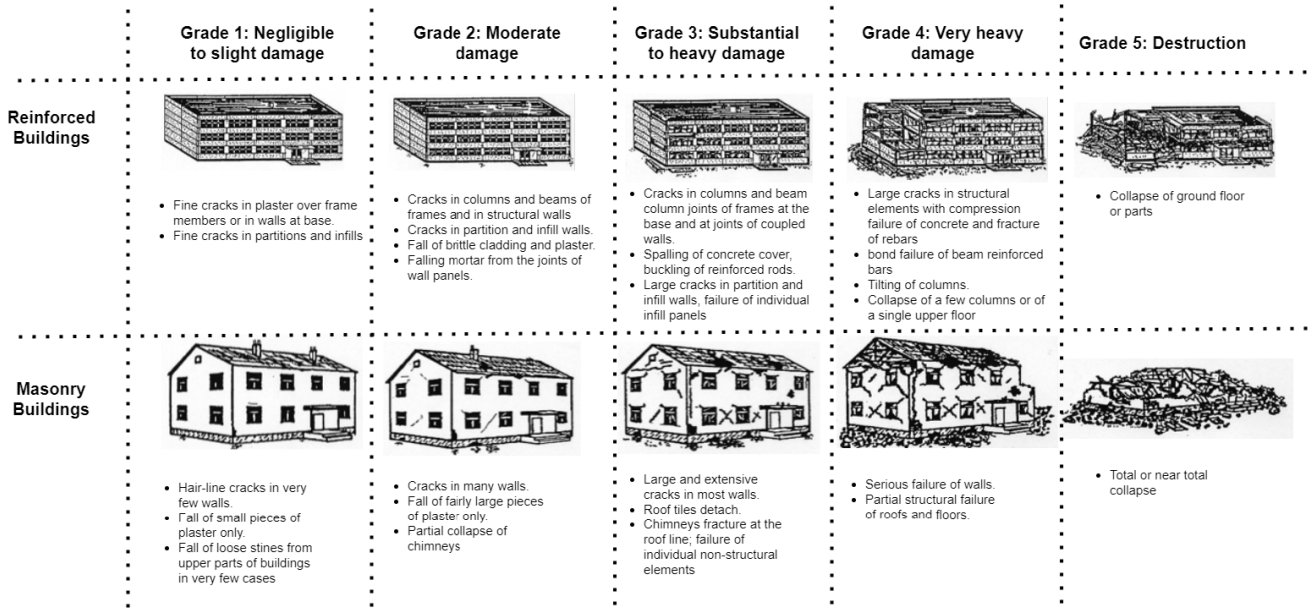


FIGURE 8. EMS-98 building damage description.

TABLE 2. The result of stratified split.

Dataset	Class	Amount	Percentage(%)
Training	Grade 1	1.535	±14.36%
	Grade 2	5.599	±52.41%
	Grade 3	1.887	±17.66%
	Grade 4	985	±9.22%
	Grade 5	677	±6.33%
Validation	Grade 1	512	±14.37%
	Grade 2	1.866	±52.40%
	Grade 3	629	±17.66%
	Grade 4	329	±9.23%
	Grade 5	225	±6.31%
Testing	Grade 1	511	±14.34%
	Grade 2	1.866	±52.40%
	Grade 3	629	±17.66%
	Grade 4	329	±9.23%
	Grade 5	226	±6.34%

split holdout technique. The stratified split is applied to equalize the class ratio on training, testing, and validation data in order to avoid missing classes where a class does not exist in a particular dataset. The result of this split is provided in Table. 2. The training data will be used to train the model, and the generalization of the model during the training phase is monitored through validation data, while testing data is only used to evaluate the final model without intervening training phase. Therefore, we ensure no leakage of testing data in the training phase.

V. PROPOSED METHOD

A. PREPROCESSING

The dataset containing the pre-earthquake and post-earthquake point cloud is extracted according to its building footprint using ArcGIS Pro Desktop software. The result of this extraction is a point cloud segmented for each building. The point cloud that was in the form of x, y, and z coordinate is transformed into DSM by projecting it into its z-axis, leaving the x and y coordinate behind and taking only the z coordinate into consideration. This results in a matrix whose width and height dimensions depend on the size of the building. each element of this matrix is the z-coordinate of the original point cloud.

This has to be done since the number of points from each building is different from the other while assessing building damage using methods such as deep learning and machine learning generally assumes that the input shape is fixed in terms of dimension. In the case of building damage assessment, point cloud should not be sampled, because sampling will cause some points to disappear as if the building was damaged even though it is not. The transformation of a point cloud into a DSM is illustrated in Fig. 9 where color implies the value of the z-axis.

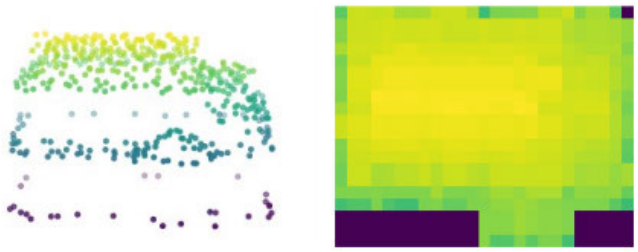


FIGURE 9. Tranformation of point cloud (left) into DSM (right).

After each building is transformed into a DSM, each DSM is resized to a fixed width and height based on the median width and median height values in the training data. For this

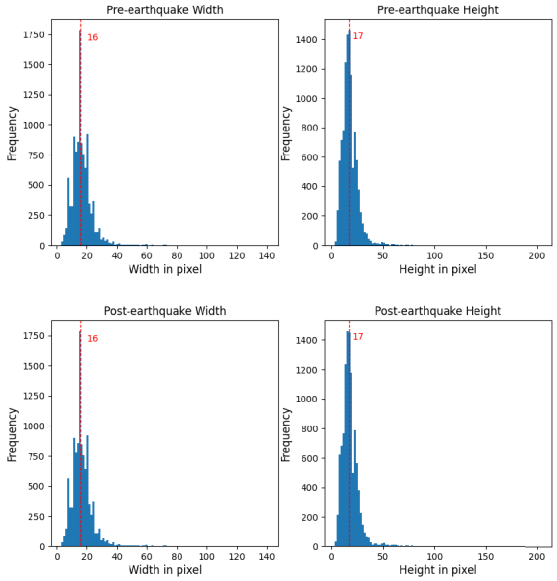


FIGURE 10. Histogram of width and height.

reason, we visualize the distribution of width and height on the training data through the histogram illustrated in Fig. 10.

As can be seen from the histogram, there is a slight positive skewness which indicates that there are extreme values in the data distribution. The existence of these extreme values will cause the average value to be large if the average value is calculated using the mean. Therefore, the average value is calculated by using the median and it turned out that the median width and height are 16 and 17 respectively. To simplify the sliding window of the convolution kernel and pooling layer, the height is rounded down to 16. Therefore all samples on training, testing, and validation data are resized to be 16×16 in width and height.

B. CLASSIFICATION METHOD

This research utilizes an SNN to classify building damage. The SNN model is modified by implementing a concatenation mechanism to concatenate the features of each convolution block into a high-dimensional vector. This is done to enrich the features of differences between the pre-earthquake and post-earthquake data since this feature seemed to be the most defining feature to detect collapsed

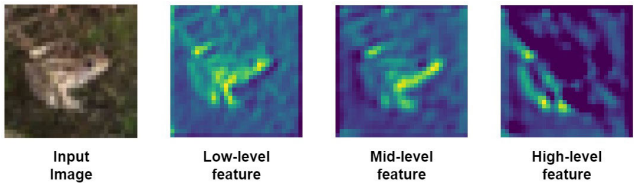


FIGURE 11. Hierarchical deep learning's feature map.

buildings [5]. As explained by Goodfellow et al. [27] in his book entitled "Deep Learning" (2016) each feature map generated by the convolution layer has a different meaning for each level (see Fig. 11). The low-level feature is extracted by the swallower layer of CNN and tends to contain minor details of the image such as lines, dots, and edges. While the high-level feature is extracted by the deeper layer of CNN and tends to contain higher abstraction of an object in the image such as shape and object part.

In this research, we aim to make a difference feature at every level of the convolution block. Therefore, the concatenation mechanism is used to gather the resulting feature vectors from each convolution block to the subtraction layer that is stacked after the last convolution block as illustrated in Fig. 12. This modification is also motivated

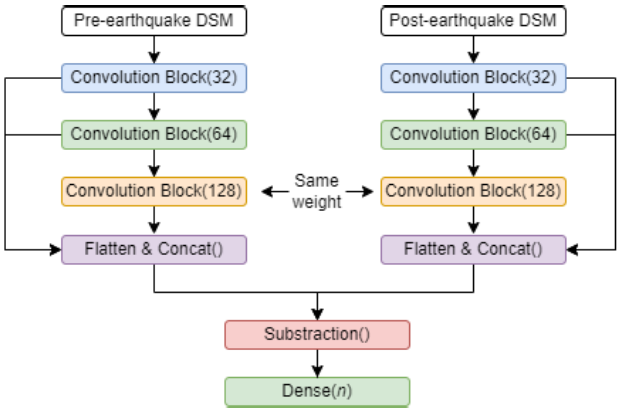


FIGURE 12. FCSNN architecture.

by Cover's Theorem [28] which states that a non-linearly separable pattern when cast into a higher dimension using a non-linear transformation is more likely to be linearly separable as illustrated in Fig. 13. This same statement is what motivates kernel function for SVM. However, this statement holds true for our model since our model uses the ReLU activation function which in itself is also a non-linear transformation

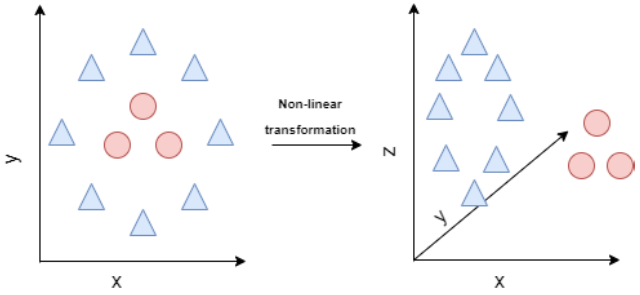


FIGURE 13. Non-linearly separable patterns become linearly separable after being cast to three dimensional using a non-linear transformation.

Therefore, we hypothesize that this modification will give better classification performance. The overall architecture of this model is illustrated in Fig. 12 and its convolution block is illustrated in Fig. 14. Each convolution layer uses 3×3 kernel with a stride of one and zero padding. Lastly, we regularized each kernel using L1 and L2 kernel regularization.

Just like a vanilla SNN, this model consists of two identical sub-networks that share the same weights and biases [22].

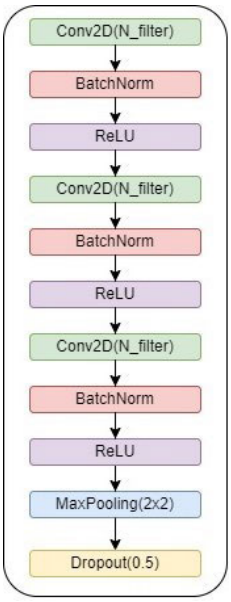


FIGURE 14. Convolution block architecture.

This is simply to extract features from two different inputs using exactly the same way. Each output from the convolution block in each sub-network is forwarded directly to the flatten and concatenation layer. This concatenation mechanism is illustrated in Fig. 15 and formulated in the equation (2).

$$\begin{aligned} \vec{u} &= \{ u_0, u_1, \ldots, u_n \} \\ \vec{v} &= \{ v_0, v_1, \ldots, v_n \} \\ \vec{w} &= \{ w_0, w_1, \ldots, w_n \} \\ \vec{x} &= [\vec{u}, \vec{v}, \vec{w}] \end{aligned} \tag{2}$$

where $\vec{u}$, $\vec{v}$, and $\vec{w}$ represent low-level feature vector, mid-level feature vector, and high-level feature vector respectively. Whereas $\vec{x}$ represents the resulting concatenation from those three vectors. The dimension of $\vec{x}$ is the total dimension of those three vectors.

On top of the concatenation layer, we stacked a subtraction layer that performs element-wise subtraction between the pre-earthquake feature vector and post-earthquake feature vector as formulated in equation (3).

$$\begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} - \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix} \tag{3}$$

where a represents the pre-earthquake feature vector, b represents the post-earthquake feature vector, and c represents the difference between the two feature vectors. Lastly, the resulting difference feature vector is fed to the last layer to perform the classification task.

C. EVALUATION

This research makes use of a confusion matrix as its evaluation metrics. A confusion matrix represents the resulting prediction summary and its label in the form of a matrix. It shows how many predictions are correct and incorrect for each class [29]. By using the confusion matrix we can understand the model performance further such as what kind of error that is frequently made by the model and which class is the most difficult for the model

The confusion matrix consists of four components as illustrated in Fig. 16. where true negative (TN) is the number of negative labels predicted as negative, false positive (FP) is the number of negative labels predicted as positive, false negative (FN) is the number of positive labels predicted as negative, and true positive (TP) is the number of positive labels predicted as positive.

1) ACCURACY

Accuracy is the number of overall true predictions divided by the total number of samples. Accuracy can be quantified using equation (4).

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

2) RECALL

Recall, also known as sensitivity, is a true positive ratio compared to the total number of positive labels. Recall can be quantified using equation (5).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5}$$

3) PRECISION

Recall, also known as sensitivity, is a true positive ratio compared to the total number of positive labels. Precision can be quantified using equation (6).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

4) F1 SCORE

F1 Score, also known as the F-measure or F-Score, is the harmonic mean between precision and recall. F1 score ranges between 0 to 1 with 0 being the worst and 1 being the best score [30]. F1 score is often used as a measurement of overall machine learning performance because this value also takes into account class imbalances in the data [30]. F1 score can be quantified using equation (7).

$$\text{F1 score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

D. MCNEMAR'S TEST

Deep learning is a model that is trained on a fairly large dataset and generally has a very large number of parameters. Deep learning is popularly known for its training process that takes quite a lot of time, it can take days or even weeks. This precludes resampling techniques from training the deep learning model many times in a single experimental run, as it will take a very long time. For this reason, this research

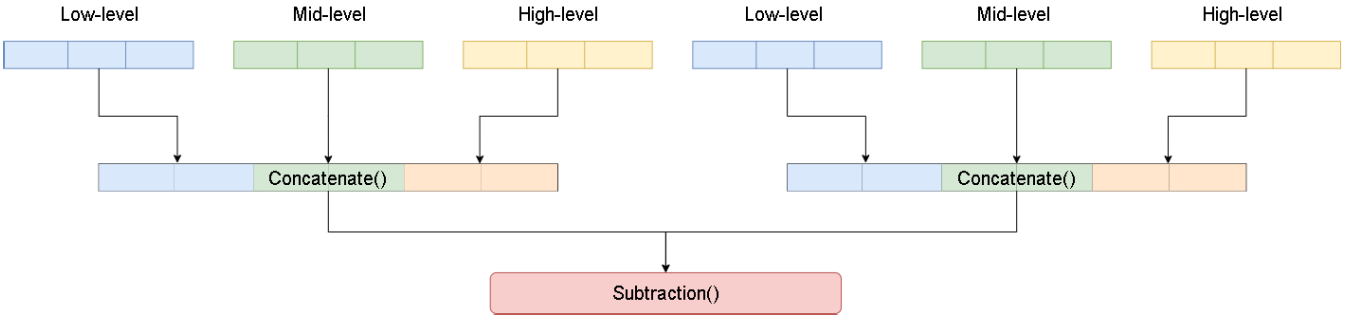


FIGURE 15. Concatenation mechanism.

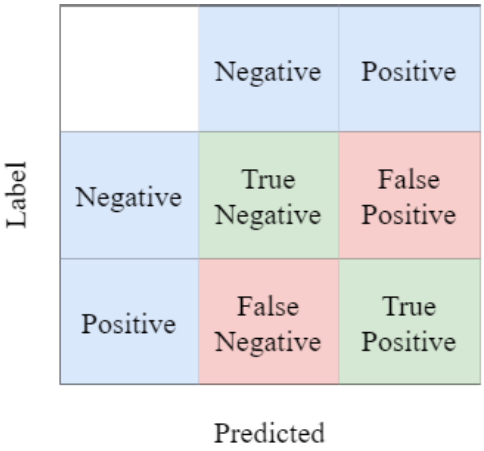


FIGURE 16. Confusion matrix.

Sample	Classifier A	Classifier B	Label
1	1	1	1
2	1	1	1
3	1	1	0
4	0	1	1
5	0	1	0

FIGURE 17. Prediction result of two different classifiers and its label.

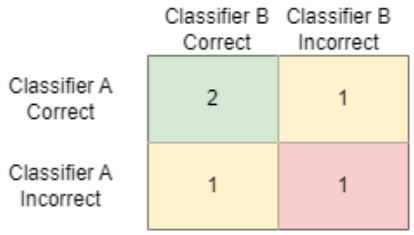


FIGURE 18. Contingency table.

used McNemar's test to compute the statistical significance between the two classification models.

McNemar's test is a suitable test for evaluating deep learning models that are large and slow to train because this test is suitable for algorithms that are executed once with an acceptable type I error [31]. It assesses the dependence of categorical data that are matched or paired [32].

McNemar's test works based on a contingency table, which in the context of a comparison of classification algorithms, the cell in this contingency table is the frequency of the two algorithms predicting the same sample as true, the frequency of the two algorithms predicting the same sample as false, and the frequency of both algorithms predicting a sample with different predictions. It allows us to see if the odds of true to false are the same for the two models [32]. For example, given the prediction results of two different classifiers, as illustrated in Fig. 17, the contingency table that will be generated is as illustrated in Fig. 18. Then McNemar's test is computed using the following equation:

$$\tilde{\chi}^2 = \frac{(|n_{0,1} - n_{1,0}| - 1)^2}{n_{0,1} + n_{1,0}} \tag{8}$$

where n is the contingency table. Then, the p-value is computed by referring to the Chi-squared distribution with one degree of freedom.

VI. RESULT AND DISCUSSION

The experiment in this research is done using Python programming language and ran in the following hardware specification:

- RAM: 32 GB.
- GPU: NVIDIA GeForce GTX 1080 Ti.
- Processor: Intel (R) Core (TM) i7-6800K CPU @ 3.40GHz.

A. IMPLEMENTATION DETAILS

In our experiment, we set the random seed for Tensorflow, Numpy, and scikit-learn to be 1234, 0, and 42 respectively. We also activate the determinism setting for the TensorFlow library. This is done to make sure that any trial we have done in every experiment uses exactly the same random behavior. Therefore, we declare that any different result in this research is caused by the model and its configuration, not the difference in the random behavior of the model.

TABLE 3. Ablation study of the first experiment scenario.

Model	Configuration				Metrics			
	Use Dense block	Balancing	Layers in each block	Use Dropout	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)
FCSNN	-	-	3 layers	yes	86.97	79.13	77.77	80.53
FCSNN	-	-	1 layer	yes	82.08	70.13	71.75	68.58
SNN	-	-	3 layers	yes	85.61	77.35	74.79	80.08
SNN	-	-	1 layer	yes	81.95	73.66	66.66	82.30
FCSNN	-	Class Weighting	3 layers	yes	82.76	75.71	66.66	87.61
FCSNN	-	Class Weighting	1 layer	yes	75.57	66.66	57.32	79.64
SNN	-	Class Weighting	3 layers	yes	78.96	73.04	60.17	92.92
SNN	-	Class Weighting	1 layer	yes	85.88	78.05	74.59	81.85
FCSNN	-	-	3 layers	-	87.65	78.98	82.60	75.66
FCSNN	-	-	1 layer	-	84.93	75.49	75.33	75.66
SNN	-	-	3 layers	-	86.56	78.14	77.97	78.31
SNN	-	-	1 layer	-	86.16	76.27	80.39	72.56
FCSNN	-	Class Weighting	3 layers	-	85.75	79.28	71.53	88.93
FCSNN	-	Class Weighting	1 layer	-	82.49	74.45	67.38	83.18
SNN	-	Class Weighting	3 layers	-	85.07	77.82	71.48	85.39
SNN	-	Class Weighting	1 layer	-	84.26	76.32	70.83	82.74
FCSNN	yes	-	3 layers	yes	85.21	77.24	73.12	81.85
FCSNN	yes	-	1 layer	yes	83.03	71.39	73.93	69.02
SNN	yes	-	3 layers	yes	85.21	75.72	76.23	75.22
SNN	yes	-	1 layer	yes	78.56	68.14	62.59	74.77
FCSNN	yes	Class Weighting	3 layers	yes	85.61	78.18	73.07	84.07
FCSNN	yes	Class Weighting	1 layer	yes	71.77	65.67	52.36	88.05
SNN	yes	Class Weighting	3 layers	yes	84.26	77.25	69.36	87.16
SNN	yes	Class Weighting	1 layer	yes	59.70	59.59	43.02	96.90
FCSNN	**yes**	**-**	**3 layers**	**-**	**87.24**	**79.47**	**78.44**	**80.53**
FCSNN	yes	-	1 layer	-	84.26	73.27	76.44	70.35
SNN	yes	-	3 layers	-	83.03	72.64	71.86	73.45
SNN	yes	-	1 layer	-	83.85	75.76	70.18	82.30
FCSNN	yes	Class Weighting	3 layers	-	85.07	77.08	72.83	81.85
FCSNN	yes	Class Weighting	1 layer	-	81.41	74.19	64.59	87.16
SNN	yes	Class Weighting	3 layers	-	84.53	76.05	72.40	80.08
SNN	yes	Class Weighting	1 layer	-	83.31	75.54	68.50	84.07

To prevent the models from overfitting, we also implemented early stopping with a patience of three.

The hyperparameter that is used in our experiment is decided through a grid search that is done for every individual model with the following hyperparameter pool:

- Optimizer: Adam, Momentum, RMSprop, Nadam, and Adamax.
- Batch size: 512, 256, 128, 64, 32, 16, and 8.
- Learning rate: 0.0001, 0.0005, 0.001, and 0.005.

The hyperparameters are chosen based on the best F1 score on the validation set. Lastly, the model with its best hyperparameter configuration is evaluated on a testing set to get the final result.

B. COMPARISON

We conducted three experiment scenarios. The first scenario is to classify only intact (Grade 1) and total collapsed (Grade 5) buildings as suggested by [6]. In the second scenario, we merged Grade 1, Grade 2, Grade 3, and Grade 4 into a single class namely non-collapsed while Grade 5 remains unchanged, this merging refers to what has been done by [1] and [5]. In the third and last scenario, we used all the 5 Grades of the EMS-98 building damage description.

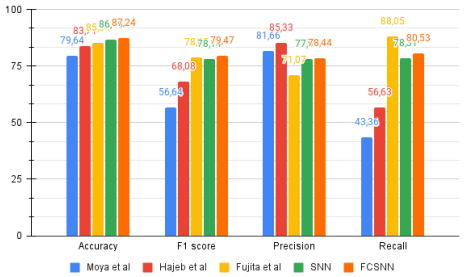


FIGURE 19. Bar chart of the first experiment scenario.

We compared our model with SNN without a concatenation mechanism. This is to study the effect of implementing a concatenation mechanism. We also compared our model with Hajeb et al.'s SVM [6], Moya et al.'s SVM [5], and Fujita et al.'s pseudo-SNN [9] point cloud-based building damage assessment model using the 2016 Kumamoto Prefecture Earthquake dataset with EMS-98 damage description. We reimplemented Fujita et al.'s model using the same dataset, environment, and treatment as ours. Although the

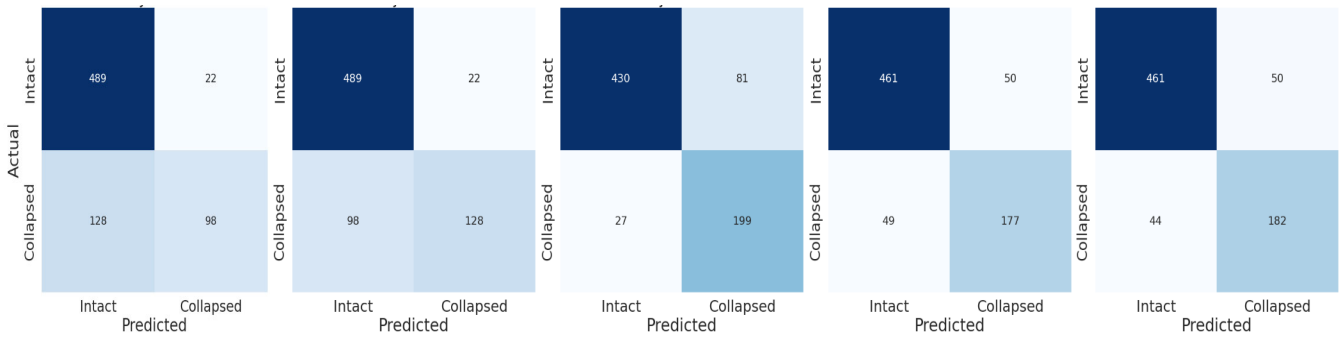


FIGURE 20. Confusion matrices of the first experiment scenario.

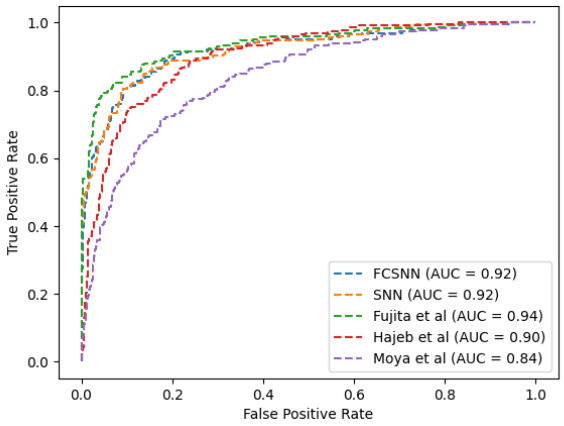


FIGURE 21. ROC curves of the first experiment scenario.

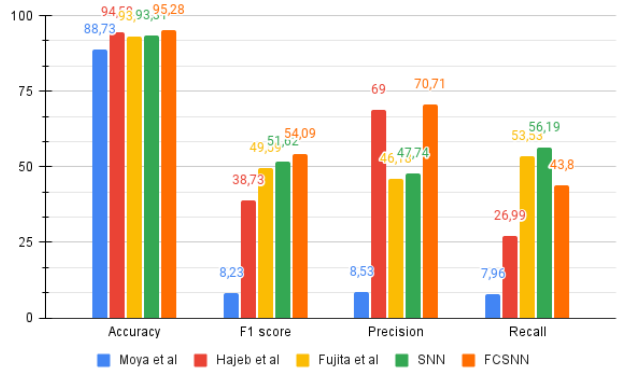


FIGURE 22. Bar chart of the second experiment scenario.

disasters used by Hajeb et al. and Moya et al. are the same as in this research, the building footprint and damage description are different from this research, therefore Hajeb et al.'s model and Moya et al.'s model are reimplemented. The reimplementation for Fujita et al.'s model is done by grid search hyperparameter with the same manner and hyperparameter pool as explained in section VI-A since their model is a deep learning-based model. Meanwhile, the reimplementation for Moya et al and Hajeb et al is done by grid searching hyperparameter with the following hyperparameter pool:

TABLE 4. Metrics of the first experiment scenario.

Model	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)	p-value
Moya et al [5]	79.64	56.64	81.66	43.36	0.001
Hajeb et al [6]	83.71	68.08	**85.33**	56.63	0.031
Fujita et al [9]	85.34	78.65	71.07	**88.05**	0.001
SNN	86.56	78.14	77.97	78.31	0.274
FCSNN	**87.24**	**79.47**	78.44	80.53	–

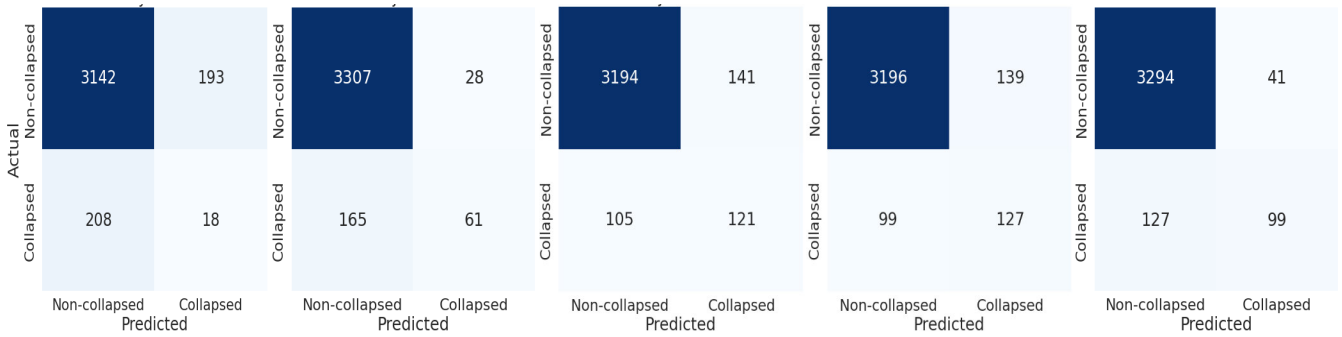
- Decision function: One-vs-rest and One-vs-one.
- Slack (C): 0.1, 1, 10, and 100
- Kernel: Linear, Polymonial, RBF, and Sigmoid.
- Gamma: Scaled, Auto, 1, 0.1, 0.01, 0.001, and 0.0001.

Additionally, we performed an ablation study to study the effect of a certain configuration on our proposed model. This includes concatenation mechanism, class weighting, dropout, adding more layers, and adding a block of dense layers to the network.

Accuracy might be the most common metric in classification, however, in the case of damage assessment where usually most of the buildings are intact, using the accuracy score alone is very deceitful [31]. Therefore, this research makes use of the confusion matrix and its four essential metrics in the classification task, which are accuracy, f1 score, precision, and recall. We also visualize the receiver operating characteristic (ROC) curve and quantified its area under the curve (AUC). Our main measurement in these experiments is the F1 score since it takes into account the imbalanced class problem [31], [33]. As an addition to that, we also measure the accuracy, precision, and recall score for the discussion purpose. Lastly, we performed hypothesis testing using McNemar's test to verify whether there is a significant difference or not. McNemar's test is a suitable test for evaluating deep learning models that are large and slow to train because this test is suitable for algorithms that are executed once with an acceptable type I error [34].

C. EXPERIMENT I

In this experiment, we performed a binary classification to classify only the intact (Grade 1) and collapsed (Grade 5) buildings. Before we compare our model against baseline



(a) Moya et al [5] (b) Hajeb et al [6] (c) Fujita et al [9] (d) SNN (e) FCSNN

FIGURE 23. Confusion matrices of the second experiment scenario.

TABLE 5. Ablation study of the second experiment scenario.

Model	Configuration				Metrics			
	Use Dense block	Balancing	Layers in each block	Use Dropout	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)
FCSNN	-	-	3 layers	yes	93.96	42.97	53.64	35.84
FCSNN	-	-	1 layer	yes	90.78	2.38	3.630	1.76
SNN	-	-	3 layers	yes	85.36	34.13	23.89	59.73
SNN	-	-	1 layer	yes	91.99	5.310	10.66	3.53
FCSNN	-	Class Weighting	3 layers	yes	86.52	39.54	27.64	69.46
FCSNN	-	Class Weighting	1 layer	yes	81.04	31.05	20.18	67.25
SNN	-	Class Weighting	3 layers	yes	90.45	45.16	35.53	61.94
SNN	-	Class Weighting	1 layer	yes	73.99	28.21	17.10	80.53
FCSNN	-	-	3 layers	-	95.39	50.89	78.70	37.61
FCSNN	-	-	1 layer	-	95.14	47.73	75.23	34.95
SNN	-	-	3 layers	-	95.02	47.16	72.47	34.95
SNN	-	-	1 layer	-	95.08	44.44	78.65	30.97
FCSNN	-	Class Weighting	3 layers	-	92.53	47.84	42.95	53.98
FCSNN	-	Class Weighting	1 layer	-	87.64	41.33	29.58	68.58
SNN	-	Class Weighting	3 layers	-	93.59	45.71	49.48	42.47
SNN	-	Class Weighting	1 layer	-	90.39	46.89	36.12	66.81
FCSNN	yes	-	3 layers	yes	88,59	42,00	31,01	65,04
FCSNN	yes	-	1 layer	yes	6,34	11,93	6,34	100
SNN	yes	-	3 layers	yes	6,34	11,93	6,34	100
SNN	yes	-	1 layer	yes	6,34	11,93	6,34	100
FCSNN	yes	Class Weighting	3 layers	yes	88,51	37,17	28,47	53,53
FCSNN	yes	Class Weighting	1 layer	yes	92,19	40,34	39,16	41,59
SNN	yes	Class Weighting	3 layers	yes	91,15	46,15	37,60	59,73
SNN	yes	Class Weighting	1 layer	yes	93,31	51,62	47,7	56,19
FCSNN	**yes**	-	**3 layers**	-	**95,28**	**54,09**	**70,71**	**43,80**
FCSNN	yes	-	1 layer	-	94,72	41,97	69,38	30,08
SNN	yes	-	3 layers	-	94,69	43,91	66,66	32,74
SNN	yes	-	1 layer	-	94,94	44,78	73,00	32,30
FCSNN	yes	Class Weighting	3 layers	-	89,97	44,13	34,14	62,38
FCSNN	yes	Class Weighting	1 layer	-	89,46	43,77	33,1	64,60
SNN	yes	Class Weighting	3 layers	-	88,59	42,00	31,01	65,04
SNN	yes	Class Weighting	1 layer	-	93,48	51,05	48,79	53,53

and [5], [6], [9], we first performed an ablation study which is provided in Table. 3. The result shows that the best configuration in terms of F1 score is to add dense blocks, having 3 layers in every block, not using dropout and class weighting with an F1 score of 79.47 as shown in row 25th in Table. 3. We also proved that implementing a concatenation mechanism (FCSNN) can enhance model performance in almost every configuration as indicated by gray rows (FCSNN) compared to white rows (SNN) in Table. 3. In the class balancing matter, it can be seen that the class weighting configuration seemed to sacrifice a lot of precision score for the sake of recall score which was not worth the trade-off, resulting in a lower F1 score compared to without balancing which tends to have higher F1 score. Our best result is obtained using a batch size of 512, a learning rate of 0.005, and an Adamax optimizer.

We then compare the result of our ablation study which was chosen based on the highest F1 score against [5], [6], [9], and our baseline model. Table. 4 provides the metrics from these five classification models which we also illustrate it using a bar chart in Fig. 19. The confusion matrices are illustrated in Fig. 20. It can be inferred that our model outperformed previous models in terms of F1 score. We achieved the highest F1 score and accuracy score. However, the highest precision is achieved by [6] with a precision score of 85.33 while the highest recall is achieved by [9].

From Fig. 20, it seemed that our model makes prediction attempt for collapsed class more often than the previous model and miss it more often too. Therefore our precision is not as high as [6]. The ROC curve is illustrated in Fig. 21 with its AUC quantified in its legend. The highest AUC score is achieved by [9] as can be seen from the confusion matrices that they have the most true positive. However, our proposed model is tied to its baseline model and outperforms [5], [6]. Using an alpha of 0.05, we can conclude that there is a significant difference in terms of classification performance between our model compared to previous models, however, the difference between our model and baseline model is not statistically significant as informed by the p-value in Table. 4.

TABLE 6. Metrics of the second experiment scenario.

Model	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)	p-value
Moya et al [5]	88.73	8.23	8.53	7.96	0.001
Hajeb et al [6]	94.58	38.73	69.00	26, 99	0.014
Fujita et al [9]	93.09	49.59	46.18	53.53	0.001
SNN	93.31	51.62	47.70	**56.19**	0.001
FCSNN	**95.28**	**54.09**	**70.71**	43.80	–

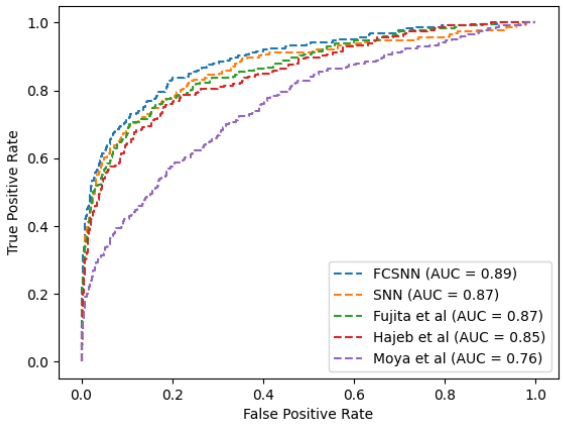


FIGURE 24. ROC curves of the second experiment scenario.

D. EXPERIMENT II

In this experiment, we performed a binary classification to classify non-collapsed (Grade 1 - Grade 4) and collapsed (Grade 5) buildings. Grade 1, Grade 2, Grade 3, and Grade 4 are merged to form a single class, namely non-collapsed while Grade 5 is left unchanged. This merging refers to what has been done by [1] and [5]. The ablation study in this experiment is provided in Table. 5. The result is quite similar to the first scenario where the highest F1 score is achieved by using a dense block, having 3 layers in each block, and not using class weighting and dropout layer. We also proved that implementing a concatenation mechanism can enhance model performance. Our best result is obtained using a batch size of 64, a learning rate of 0.001, and a Nadam optimizer.

In this experiment, we are facing an extremely imbalanced class with a ratio of 93.66%: 6.33% for non-collapsed and collapsed respectively. These models seemed to struggle in detecting positive class, hence the high accuracy but low F1 score. As a result, these models are capable of detecting negative classes easily. This is a very common phenomenon when a classification model is trained on an imbalanced dataset, the model will make prediction attempts for the majority class more often than the minority class. The implementation of class weighting indeed increases the recall score. However, just like in the previous scenario, it sacrifices a lot of precision scores for a slight improvement in recall score which is not worth the trade-off.

The comparison between our model and the previous models is provided in Table. 6 and Fig. 22. Its confusion matrices are illustrated in Fig. 23. It can be inferred that our model outperformed previous models in terms of F1 score. Although our model is not the one that captures the positive class the most, however, our model managed to minimize the false positive rate which leads to a higher AUC score as can be seen through the ROC curve in Fig. 24. The p-value informed that there was a significant difference in terms of classification performance between our model and previous models.

E. EXPERIMENT III

In this experiment, we performed a multiclass classification to classify the five Grades of EMS-98 building damage description. Grade 1, Grade 2, Grade 3, Grade 4, and Grade 5 are left unchanged. Each metric in this experiment is averaged throughout all the classes. We adjusted our model for the multiclass classification problem. The output layer consists of five neurons with softmax activation function while the loss function is changed to categorical cross-entropy. The result of the ablation study in this experiment is provided in Table. 7.

The result of this experiment is somewhat different from those in the first and second experiments. The highest F1 score is achieved without dense block and by using class weighting and dropout. It is no surprise that we get different configurations since this is a multiclass classification which is a very different task from those in the first and second experiments where we performed binary classification. We also proved that in almost any configuration, implementing a concatenation mechanism can enhance the model's performance. Our best result is obtained using a batch size of 16, a learning rate of 0.0001, and an Adam optimizer.

The comparison between our model and the previous models is provided in Table. 8 and Fig. 25. Its confusion

TABLE 7. Ablation study of the third experiment scenario.

Model	Configuration				Metrics			
	Use Dense block	Balancing	Layers in each block	Use Dropout	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)
FCSNN	-	-	3 layers	yes	60.29	37.82	35.53	41.41
FCSNN	-	-	1 layer	yes	55.57	25.29	21.42	30.95
SNN	-	-	3 layers	yes	59.33	36.75	34.63	40.45
SNN	-	-	1 layer	yes	55.54	32.46	31.57	36.04
FCSNN	**-**	**Class Weighting**	**3 layers**	**yes**	**42.57**	**40.64**	**39.27**	**47.29**
FCSNN	-	Class Weighting	1 layer	yes	34.45	33.03	32.11	39.25
SNN	-	Class Weighting	3 layers	yes	37.96	37.40	36.82	45.76
SNN	-	Class Weighting	1 layer	yes	32.96	34.14	36.29	44.47
FCSNN	-	-	3 layers	-	60.76	38.30	42.59	40.76
FCSNN	-	-	1 layer	-	57.39	33.61	36.68	34.66
SNN	-	-	3 layers	-	60.85	39.29	42.67	41.58
SNN	-	-	1 layer	-	59.42	36.74	38.16	38.19
FCSNN	-	Class Weighting	3 layers	-	40.66	39.67	39.06	47.39
FCSNN	-	Class Weighting	1 layer	-	38.78	36.09	35.71	45.30
SNN	-	Class Weighting	3 layers	-	38.41	38.02	37.92	48.16
SNN	-	Class Weighting	1 layer	-	39.00	36.89	35.99	44.87
FCSNN	yes	-	3 layers	yes	60.20	38.23	35.83	41.98
FCSNN	yes	-	1 layer	yes	54.31	24.62	20.23	31.61
SNN	yes	-	3 layers	yes	57.14	25.39	24.82	28.77
SNN	yes	-	1 layer	yes	56.02	23.75	23.25	27.23
FCSNN	yes	Class Weighting	3 layers	yes	43.89	38.92	37.09	45.24
FCSNN	yes	Class Weighting	1 layer	yes	29.68	30.23	34.81	41.45
SNN	yes	Class Weighting	3 layers	yes	41.61	39.02	38.18	43.13
SNN	yes	Class Weighting	1 layer	yes	36.16	31.70	38.21	37.91
FCSNN	yes	-	3 layers	-	60.82	38.61	39.23	40.42
FCSNN	yes	-	1 layer	-	60.34	37.87	38.21	39.99
SNN	yes	-	3 layers	-	57.90	38.74	41.27	39.48
SNN	yes	-	1 layer	-	60.40	38.14	38.50	40.08
FCSNN	yes	Class Weighting	3 layers	-	39.06	39.80	39.40	48.55
FCSNN	yes	Class Weighting	1 layer	-	38.69	37.58	36.78	46.33
SNN	yes	Class Weighting	3 layers	-	43.02	40.47	39.97	47.37
SNN	yes	Class Weighting	1 layer	-	39.76	38.72	37.95	47.15

TABLE 8. Metrics of the third experiment scenario.

Model	Accuracy(%)	F1 score(%)	Precision(%)	Recall(%)	p-value
Moya et al [5]	42.17	22.31	27.91	25.06	0.762
Hajeb et al [6]	52.70	34.91	**42.95**	33.38	0.001
Fujita et al [9]	**60.74**	39.15	38.74	43, 14	0.001
SNN	43.02	40.47	39.97	**47.37**	0.001
FCSNN	42.57	**40.64**	39.27	47.29	–

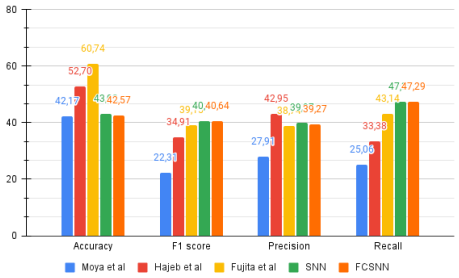


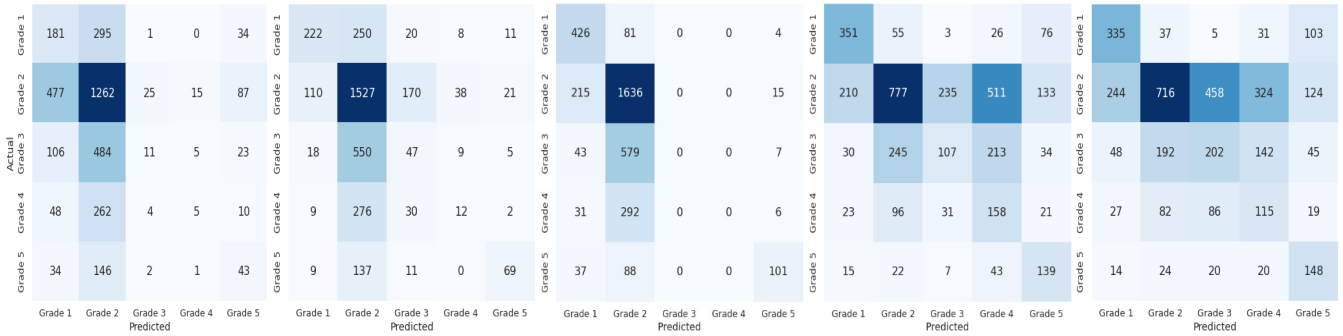
FIGURE 25. Bar chart of the third experiment scenario.

matrices are illustrated in Fig. 26 while ROC curves are illustrated in Fig. 27. The highest accuracy score is achieved by [9], however, their model has a lower recall score which results in a lower F1 score.

The confusion matrices and ROC curves show us that there are classes that are hard to detect. They are Grade 3 and Grade 4. We believe this is due to the characteristic of those classes are mainly located in the wall (see Fig. 8) which is hard to capture using Airborne LiDAR since the sensor is flying, resulting in a point cloud that doesn't contain any information about the wall as can be seen in Fig. 9. Moreover, the DSM method only takes into account the z-axis of the point cloud. Grade 2 is easy to detect since this class is the majority class. Grade 1 and grade 5 are somewhat easier because these classes are very well presented using DSM. The p-value informed us that there is a significant difference between our model against Hajeb et al and Fujita et al, meanwhile there is no significant difference between our model against Moya et al as can be seen form Table. 7.

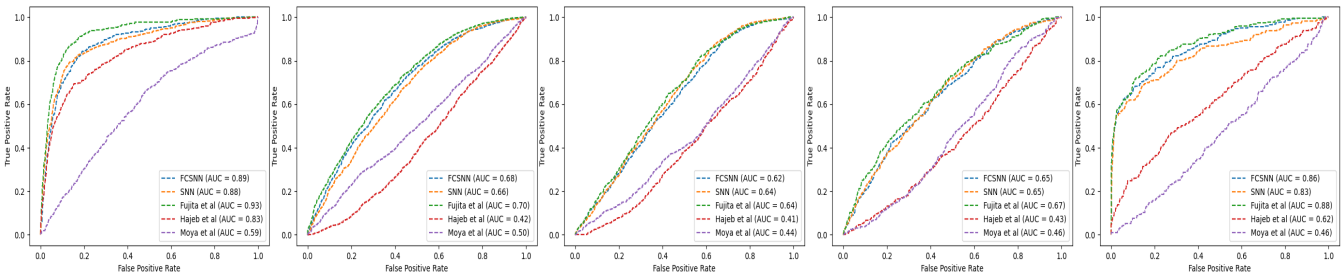
F. GENERALIZATION

The FCSNN model has better performance because the output layer receives a high-dimensional concatenated vector. This proves what Cover stated that a non-linearly separable pattern after increasing its dimensions through a non-linear



(a) Moya et al [5] (b) Hajeb et al [6] (c) Fujita et al [9] (d) SNN (e) FCSNN

FIGURE 26. Confusion matrices of the third experiment scenario.



(a) Grade 1 (b) Grade 2 (c) Grade 3 (d) Grade 4 (e) Grade 5

FIGURE 27. ROC curves of the third experiment scenario.

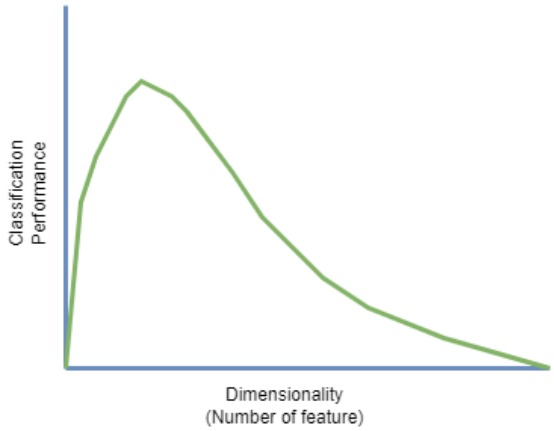


FIGURE 28. Curse of dimensionality.

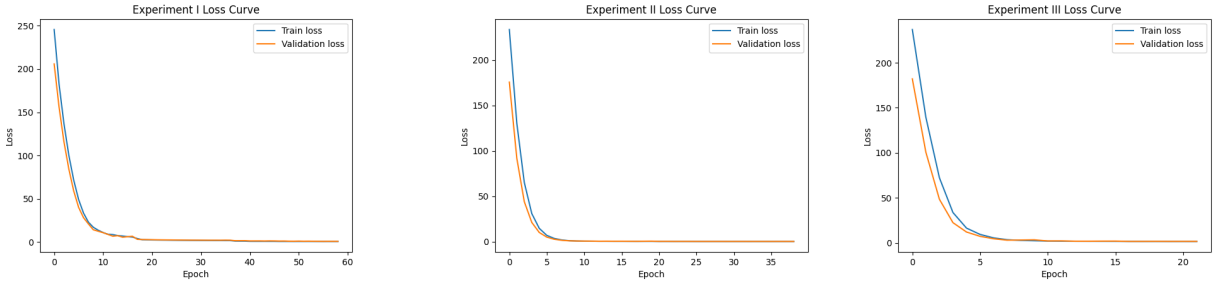
transformation will make the pattern more linearly separable is true [28]. However, adding these dimensions will not always have a good effect on the model's performance.

Hughes stated that a classification model with initially low dimensional data, its accuracy can be increased by adding dimensions, but if the dimension is too high the accuracy of the classification model will tend to decrease as illustrated in Fig. 28. This is also known as the Hughes phenomenon or the curse of dimensionality where in high dimensions, the model will find it easier to find a hyperplane to separate each class in the training data even if the data has a lot of noise, so the resulting model tends to experience overfitting problem [35], [36].

TABLE 9. Model's generalization.

First experiment	
Dataset	Binary crossentropy loss
Training set	0, 305165
Validation set	0, 365924
Testing set	0, 349004
Generalization on validation set	−0, 060759
Generalization on testing set	−0, 043839
Second experiment	
Dataset	Binary crossentropy loss
Training set	0, 155560
Validation set	0, 168771
Testing set	0, 159026
Generalization on validation set	−0, 013211
Generalization on testing set	−0, 003465
Third experiment	
Dataset	Categorical crossentropy loss
Training set	1, 420278
Validation set	0, 168771
Testing set	0, 159026
Generalization on validation set	1, 251507
Generalization on testing set	1, 261252

For that reason, we discuss the generalization of the proposed model chosen based on the highest f1 score from the three experimental scenarios that have been carried out. Table. 9 provides the final model's loss for training data, the model's loss for validation data, and the model's loss for testing data. The loss curve is provided in Fig. 29. We quantify



(a) Experiment I (b) Experiment II (c) Experiment III

FIGURE 29. Loss curves.

generalization by calculating the loss differential between training and testing. It is done by subtracting the training loss and testing loss.

The loss differential of the model on validation data and testing data in the three scenarios is not huge, which means that the loss in the training data is not much different from the loss in the validation data and testing data. Therefore, we state that this model does not experience overfitting problems. This might happen because the resulting dimensions are not too high to cause the model to experience an overfitting problem. The loss differential of the model in the third experiment has a positive value indicating that the performance on validation data and testing data is better than the performance on training data. This is due to too many regularizations applied for the best configurations found in experimental scenario 3, bearing in mind that the best results in experimental scenario 3 are obtained by implementing class weights and dropouts.

VII. CONCLUSION

This research develops a classification model for building damage assessment based on pre-earthquake point cloud and post-earthquake point cloud. The problem we discussed in this paper is the usage of handcrafted features which require human expertise. We automate this by using an end-to-end connected model. The basis of our proposed model is an SNN. We apply some modifications to an SNN to enhance its classification capability.

We conducted three experiment scenarios where we outperformed our comparison models and achieved an F1 score of 79.47%, 54.09%, and 40.64% for experiment I, experiment II, and experiment III respectively. In terms of accuracy, we achieved an accuracy score of 87.24%, 95.28%, and 42.57% for experiment I, experiment II, and experiment III respectively. Indeed, this result can still be improved by implementing certain approaches to improve the performance of this model. This research is an initial research on the FCSNN. Therefore, there are still many things that can be improved and further worked on in the future.

For future work, we would like to consider the use of Voxel as a replacement for DSM to improve our results. We also plan to conduct some research related to this topic using different sub-network architectures for the FCSNN. The most important thing in this research is introducing a novel architecture based on an SNN. We also have a plan to bring this model into other fields such as sequence embedding and compare our result with Zheng et al.'s SENSE [37]. The combination of our idea and SENSE also interests us. Another field that caught our interest is image analysis such as patch matching by Hanif et al. [38] which uses enhanced two-channel and SNN on the UBC image patch benchmark dataset [39]. For now, we hope this research can be useful as a foundation for future research.

REFERENCES

[1] M. Ji, L. Liu, R. Du, and M. F. Buchroithner, "A comparative study of texture and convolutional neural network features for detecting collapsed buildings after earthquakes using pre- and post-event satellite imagery," *Remote Sens.*, vol. 11, no. 10, p. 1202, May 2019.

[2] A. W. Coburn, R. J. S. Spence, and A. Pomonis, "Factors determining human casualty levels in earthquakes: Mortality prediction in building collapse," in *Proc. 10th World Conf. Earthquake Eng.*, 1992, pp. 5989–5994.

[3] R. Borg, M. Indirli, T. Rossetto, and L. Kouris, "L'aquila earthquake April 6th, 2009: The damage assessment methodologies," in *Proc. COST Action C26 Final Conf.*, 2010.

[4] R. Das and S. Hanaoka, "An agent-based model for resource allocation during relief distribution," *J. Humanitarian Logistics Supply Chain Manag.*, vol. 4, no. 2, pp. 265–285, Oct. 2014.

[5] L. Moya, F. Yamazaki, W. Liu, and M. Yamada, "Detection of collapsed buildings from LiDAR data due to the 2016 Kumamoto earthquake in Japan," *Natural Hazards Earth Syst. Sci.*, vol. 18, no. 1, pp. 65–78, Jan. 2018.

[6] M. Hajeb, S. Karimzadeh, and M. Matsuoka, "SAR and LiDAR datasets for building damage evaluation based on support vector machine and random forest algorithms—A case study of Kumamoto earthquake, Japan," *Appl. Sci.*, vol. 10, no. 24, p. 8932, Dec. 2020.

[7] W. Zhai, H. Shen, C. Huang, and W. Pei, "Building earthquake damage information extraction from a single post-earthquake PolSAR image," *Remote Sens.*, vol. 8, no. 3, p. 171, Feb. 2016.

[8] B. Höller, A. Mossel, and H. Kaufmann, "Automatic object annotation in streamed and remotely explored large 3D reconstructions," *Comput. Vis. Media*, vol. 7, no. 1, pp. 71–86, Mar. 2021.

[9] A. Fujita, K. Sakurada, T. Imaizumi, R. Ito, S. Hikosaka, and R. Nakamura, "Damage detection from aerial images via convolutional neural networks," in *Proc. 15th IAPR Int. Conf. Mach. Vis. Appl. (MVA)*, May 2017, pp. 5–8.

[10] M. Shafique, M. van der Meijde, and M. A. Khan, "A review of the 2005 Kashmir earthquake-induced landslides; from a remote sensing prospective," *J. Asian Earth Sci.*, vol. 118, pp. 68–80, Mar. 2016.

[11] Y. Dong, Q. Li, A. Dou, and X. Wang, "Extracting damages caused by the 2008 Ms 8.0 Wenchuan earthquake from SAR remote sensing data," *J. Asian Earth Sci.*, vol. 40, no. 4, pp. 907–914, Mar. 2011.

[12] S. Xie, J. Duan, S. Liu, Q. Dai, W. Liu, Y. Ma, R. Guo, and C. Ma, "Crowdsourcing rapid assessment of collapsed buildings early after the earthquake based on aerial remote sensing image: A case study of Yushu earthquake," *Remote Sens.*, vol. 8, no. 9, p. 759, Sep. 2016.

[13] H. Miura, T. Aridome, and M. Matsuoka, "Deep learning-based identification of collapsed, non-collapsed and blue tarp-covered buildings from post-disaster aerial images," *Remote Sens.*, vol. 12, no. 12, p. 1924, Jun. 2020.
[14] M. He, Q. Zhu, Z. Du, H. Hu, Y. Ding, and M. Chen, "A 3D shape descriptor based on contour clusters for damaged roof detection using airborne LiDAR point clouds," *Remote Sens.*, vol. 8, no. 3, p. 189, Feb. 2016.
[15] A. S. Al-Waisy, R. Qahwaji, S. Ipson, S. Al-Fahdawi, and T. A. M. Nagem, "A multi-biometric iris recognition system based on a deep learning approach," *Pattern Anal. Appl.*, vol. 21, pp. 783–802, Aug. 2018.
[16] J. Li, X. Zhou, S. Chan, and S. Chen, "Object tracking using a convolutional network and a structured output SVM," *Comput. Vis. Media*, vol. 3, no. 4, pp. 325–335, Dec. 2017.
[17] X. Liu, Y. Zhang, F. Bao, K. Shao, Z. Sun, and C. Zhang, "Kernel-blending connection approximated by a neural network for image classification," *Comput. Vis. Media*, vol. 6, no. 4, pp. 467–476, Dec. 2020.
[18] M. Sahni, R. Sahni, and J. M. Merigo, *Neural Networks, Machine Learning, and Image Processing: Mathematical Modeling and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, 2022.
[19] J. G. Nicholls, A. R. Martin, P. A. Fuchs, D. A. Brown, M. E. Diamond, and D. A. Weisblat, *From Neuron to Brain*, 5th ed. Sunderland, MA, USA: Sinauer Associates, 2012, pp. 3–43.
[20] R. A. Ibrahim, A. H. Elsheikh, M. E. A. Elaziz, and M. A. A. Al-qaness, "Chapter one—Basics of artificial neural networks," in *Artificial Neural Networks for Renewable Energy Systems and Real-World Applications*, A. H. Elsheikh and M. E. Abd Elaziz, Eds. Academic, 2022, pp. 1–10.
[21] M. Mohseni-Dargah, Z. Falahati, B. Dabirmanesh, P. Nasrollahi, and K. Khajeh, "Chapter 12—Machine learning in surface plasmon resonance for environmental monitoring," in *Artificial Intelligence and Data Science in Environmental Sensing*, M. Asadnia, A. Razmjou, and A. Beheshti, Eds. Cambridge, MA, USA: Academic Press, 2022, pp. 269–298.
[22] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a 'Siamese' time delay neural network," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 7, p. 25, Aug. 1993.
[23] D. Chicco, "Siamese neural networks: An overview," in *Artificial Neural Networks*, H. Cartwright, Ed. New York, NY, USA: Springer, 2021, pp. 73–94.
[24] A. Mehmood, M. Maqsood, M. Bashir, and Y. Shuyuan, "A deep Siamese convolution neural network for multi-class classification of Alzheimer disease," *Brain Sci.*, vol. 10, no. 2, p. 84, Feb. 2020.
[25] Pre-Kumamoto Earthquake, *Rupture LiDAR Scan. Distributed by OpenTopography*. San Diego, CA, USA: Univ. of California, Apr. 2016, doi: 10.5069/G9XP7303.
[26] Post-Kumamoto Earthquake, *Rupture LiDAR Scan. Distributed by OpenTopography*. San Diego, CA, USA: Univ. of California, Apr. 2016. [Online]. Available: https://doi.org/10.5069/G9SX6B9T
[27] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[28] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electron. Comput.*, vol. EC-14, no. 3, pp. 326–334, Jun. 1965.
[29] R. Pandey, S. K. Khatri, N. K. Singh, and P. Verma, *Artificial Intelligence and Machine Learning for EDGE Computing*, vol. 125 London, U.K.: Elsevier, 2022.
[30] I. H. Witten, E. Frank, and M. A. Hall, "Chapter 5—Credibility: Evaluating what's been learned," in *Data Mining: Practical Machine Learning Tools and Techniques*, I. H. Witten, E. Frank, and M. A. Hall, Eds., 3rd ed. Boston, MA, USA: Morgan Kaufmann, 2011, pp. 147–187.
[31] H. Narasimhan, W. Pan, P. Kar, P. Protopapas, and H. G. Ramaswamy, "Optimizing the multiclass F-measure via biconcave programming," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 1101–1106.
[32] R. H. Riffenburgh, "Chapter 15—Tests on categorical data," in *Statistics in Medicine*, R. H. Riffenburgh, Ed. 2nd ed. Burlington, NJ, USA: Academic Press, 2006, pp. 241–279.
[33] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach*. Beijing, China: O'Reilly, 2017.
[34] T. Kavzoglu, "Chapter 33—Object-oriented random forest for high resolution land cover mapping using quickbird-2 imagery," in *Handbook of Neural Computation*, P. Samui, S. Sekhar, and V. E. Balas, Eds. Cambridge, MA, USA: Academic Press, 2017, pp. 607–619.
[35] G. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Trans. Inf. Theory*, vol. IT-14, no. 1, pp. 55–63, Jan. 1968.
[36] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2022.
[37] W. Zheng, L. Yang, R. J. Genco, J. Wactawski-Wende, M. Buck, and Y. Sun, "SENSE: Siamese neural network for sequence embedding and alignment-free comparison," *Bioinformatics*, vol. 35, no. 11, pp. 1820–1828, Jun. 2019.
[38] M. S. Hanif, "Patch match networks: Improved two-channel and Siamese networks for image patch matching," *Pattern Recognit. Lett.*, vol. 120, pp. 54–61, Apr. 2019.
[39] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 43–57, Jan. 2011.

MGS M. LUTHFI RAMADHAN received the bachelor's degree from the Faculty of Computer Science, University of Sriwijaya, and the master's degree from the Faculty of Computer Science, University of Indonesia. He is currently a Research Assistant with the Faculty of Computer Science, University of Indonesia. His research interests include deep learning, pattern recognition, and computer vision.

GRAFIKA JATI received the B.S. and M.Sc. degrees in computer science from the University of Indonesia, in 2014 and 2016, respectively. He is currently pursuing the Ph.D. degree in automotive engineering for intelligent mobility with the University of Bologna. From 2017 to 2022, he was a Research Assistant with the Faculty of Computer Science, University of Indonesia. His research interests include artificial intelligence, visual object tracking, and autonomous robots.

WISNU JATMIKO (Senior Member, IEEE) received the B.S. degree in electrical engineering and the M.Sc. degree in computer science from the University of Indonesia, Depok, Indonesia, in 1997 and 2000, respectively, and the Dr.-Eng. degree from Nagoya University, Japan, in 2007. He is currently a Full Professor with the Faculty of Computer Science, University of Indonesia. His research interests include autonomous robots, optimization, real-time traffic monitoring systems, machine learning, and artificial intelligence. He is also the Chairperson of the IEEE Indonesia Section, from 2019 to 2020.