A disaster classification application using convolutional neural network by performing data augmentation

Mummaneni Sobhana[1], Smitha Chowdary Chaparala[1], Devaganugula N. V. S. L. S. Indira[2], Konduru Kranthi Kumar[3]

[1]Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, India
[2]Department of Computer Science and Engineering, Malla Reddy College, Haiderabad, India
[3]Department of Information Technology, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru, India
[4] Department of Computer Science and Engineering, Dhanekula Institute of Engineering and Technology, Ganguru, India

Article Info

Article history:

Received Aug 8, 2021
Revised Jun 18, 2022
Accepted Jul 4, 2022

Keywords:

Computer vision
Convolutional neural networks
Deep learning
Image processing
Natural disasters

ABSTRACT

Natural disasters are catastrophic events and cause havoc to human life. These events occur in the most unpredictable times and are beyond human control. The aftermath of the disasters is devastating ranging from loss of life to relocation of large groups of the population. With the development in the domains of computer vision (CV) and Image processing, machine learning and deep learning models can integrate images and perform predictions. Deep learning techniques employ many robust techniques and provide significant results even in the case of images. The detection of natural disasters without human intervention requires the help of deep learning techniques. The project aims to employ a multi-layered convolutional neural network (CNN) organization to classify the images related to natural disasters related to earthquakes, floods, cyclones, and wildfires.

Corresponding Author:

Mummaneni Sobhana
Department of Computer Science and Engineering,
Velagapudi Ramakrishna Siddhartha Engineering College
Kanuru, Vijayawada, Andhra Pradesh-5200007, India
Email: sobhana@vrsiddhartha.ac.in

1. INTRODUCTION

Natural disasters disturb the human life system and destroy the settlements of human habitats and even lead to a long-lasting change in the world. They are unavoidable, and their occurrence affects the financial system, habitats, and also human life. The aftermath of disasters leaves humans in miserable situations, and sometimes the devastating effects are not detectable. Rescue operations cannot take place in most places, and victims cannot be identified due to the wreckage and the commotion caused due to it. Floods are caused due to overflow of water bodies such as rivers that are close to human settlements. Tonini *et al.* [1] employed a random forest technique to identify regions susceptible to wildfires. Islam *et al.* [2] used a support vector machine (SVM) and random forest (RF) technique to map flood-prone areas. Many techniques have been implemented to predict disasters like wildfires.

The increase in human settlements has caused disturbance to the ecosystem causing these disasters. Adding to it is the increasing pollution and emission of greenhouse gases affecting the whole planet. Underdeveloped countries cannot afford the cost of restoration of infrastructure after being hit by a disaster. Loss of life is a serious threat posed by natural disasters, they are unavoidable and, in many cases, undetectable.

With the increase in surveillance devices, data is within the reach of everyone. Computer vision and machine learning techniques are now being integrated into various industries. Many techniques have been employed by various scientists to observe and classify natural disasters to overcome losses of lives, but its detection still faces problems due to the complexity and varied shapes, sizes, and quality of images. Many studies have proposed techniques for natural disaster classification using neural networks and image processing.

Deep learning has recently seen remarkable growth in a variety of fields. As opposed to traditional machine learning strategies, deep learning and computer vision has a robust learning capacity and can improve data set utilization for feature extraction. Images have a lot of irregularities and lack uniformity by nature. The processing of these images is thus a challenge that entails a series of steps.

Neural networks provide a multilayered architecture for extracting features that are complex to be extracted using conventional machine learning techniques. Convolutional neural networks (CNN) are frequently implemented for image-related applications. It can perform image processing without much complexity. Tang *et al.* [3] have shown that the feature extraction using CNN highly efficient and CNNs can identify features even from noisy data. Usage of neural networks for the detection of natural disasters has gained prominence in recent decades. Li *et al.* [4] proposed a CNN named "Wildfire smoke dilated dense net for detection of wildfires from images". Mangalathu and Burton [5] proposed a deep learning method using long short-term memory (LSTM) for analyzing the effect of earthquakes on building clusters after the impact of earthquakes.

Aamir *et al.* [6] proposed a model which works in two blocks: The first block consists of a neural network for identifying and detecting the event of a natural disaster. This block ensures to accurately classify the images into the corresponding disaster. The next block of the CNN is used to further analyze the intensity of the disaster. For this, images are provided for training the model to identify the intensity similarity for foreign images. For this, the model makes use of several parameters and filters. By following this methodology, the authors were able to obtain a competitive accuracy of 90.92%. This is comparatively equal to the accuracies of many state-of-the-art algorithms in existence. On the other hand, the paper does not address the overcoming of imbalanced datasets. Panigrahi *et al.* [7] proposed a CNN architecture comprising three-layered structures (Two convolutional-pooling and one fully connected). The metrics considered here are accuracy and loss. The uniform size set for the model input is 64864x3 signifying a colored image. The kernel size is 3×3. The max-pooling region size is 2×2. The third layer consists of 128 rectified linear unit (ReLU) neurons which are fully connected. The loss function considered in the system is binary cross-entropy. To avoid overfitting, dropout was used to dispose of some neurons from the system.

Obaid *et al.* [8] address the challenges that CNN needs to deal with. The paper discusses two augmentation techniques: oversampling and warping. The geometric techniques such as flipping of images, color space identification, improvement by cropping, rotation, translation, and noise injection were discussed. Color space transformations, random erasing and, generative adversarial networks (GANs) were some other techniques discussed.

The research proposed by Patil and Banyal [9] entitled "Techniques of deep learning for image recognition" has studied the various approaches and techniques available for deep learning in image recognition. The various learning modes such as supervised, unsupervised with various data inputs have yielded results where CNN has been proved as the optimal technique for image recognition. Albawi *et al.* [10] have proposed a research paper titled "Understanding of a convolutional neural network" which illustrates and explains all the important issues and parameters pertaining to CNN, how these elements work, and stated the parameters that affect CNN efficiency. The paper started with an introduction to neural network concepts and dive deep down to the heart of execution on how the convolutional neural network works.

According to the work proposed by Chauhan *et al.* [11] CNN when provided with the required number of parameters and filters can be used to implement humongous datasets with several parameters. The images provided in the 2-dimensional matrix format can be easily convolved by the filters of CNN, to provide the best results. Jmour *et al.* [12] proposed a system describing an approach based on using convolutional neural networks (CNN) for traffic sign identification systems. It also provides the results of applying this CNN to learn features and classify color images [13].

2. METHODS

2.1. Dataset

The images were collected from various online resources available. The dimensions of the image can range from any height to width. Color images red, green, and blue (RGB) images belonging to all the disasters were considered [14]. The existing dataset consists of 900 images belonging to each of the classes i.e., 3,600 images in total. The data is divided for computation in such a way that 800 images of each class are used for training and 100 images of each class are saved for testing. Initially, images in the dataset are of various sizes thus resulting in various numbers of pixels. Table 1 shows the data distribution.

Table 1. Data division

Category	Number of images
Cyclone	900
Earthquake	900
Flood	900
Wildfire	900
Total	3,600

2.2. Method

2.2.1. Data augmentation and pre-processing

In many cases, the data available may not be diverse enough to effectively train the model [15]. In such cases adding versatile images with different features such as color, and orientation [16]. can help in building a model free from losses. Appending new images to the existing dataset can sometimes be difficult. Also, obtaining new images may not be feasible [17]. In cases like natural disasters, the events are unpredictable, and capturing the images during the occurrence of the event is practically infeasible [18]. In such cases appending data to the existing data can be done using the available data itself. Data augmentation is the technique used to increase the sample space required for any model [19]. To obtain reliable results, the model requires to be trained on several instances of data [20]. This requirement cannot be always satisfied due to various reasons such as commercial inviability and infeasibility. If all possible instances are provided to the model, then the model trains to become more generalized rather than being restricted to training data alone [21]. The Keras deep learning neural network library provides the capability to fit models using image data augmentation via the ImageDataGenerator class. From the original dataset 400 images belonging to each class were used to generate 5 new instances. Figure 1 shows the output after data augmentation of a single image.

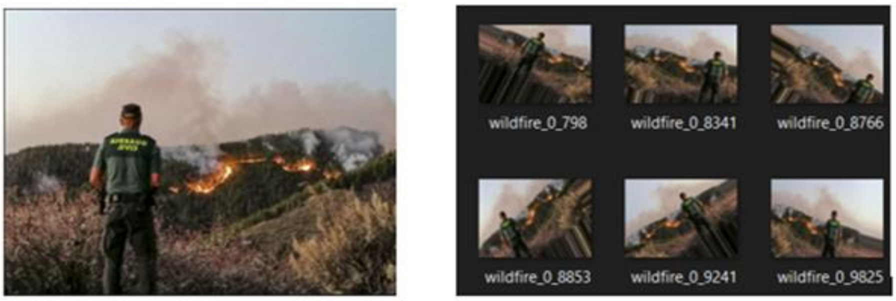

Figure 1. Data augmentation output

A total of 2,000 images were generated for each class. From the remaining 500 images, 400 were added to the previous 2,000 and the resultant 2400 images were treated as training data. The remaining 100 images in each class were used for testing purposes. The training set contains 9,600 images, where 20%, i.e., 1920 images were treated as the validation set. The reason for using a validation set is to provide an estimate which is not restricted to training data but remains unbiased to estimate the performance of the final model [22].

The next step in data preparation involves pre-processing the data. The received data usually contains a lot of noise that comes from various sources [23]. To provide them for training the neural network model, the entire dataset needs to follow a uniform standardization and need to go through a careful preprocessing step. Preprocessing is used to improve the quality of the images and reduce the complexity and thereby increase the accuracy of the final model. Performing different sequences of steps for different images is not easy to be designed, so a more generalized algorithm that converts the entire image into a format acceptable by the model is important.

Another constraint that exists in various neural network algorithms, such as CNN, is the need to see that all the images belong to the same size categories [24]. This ensures that the model learns the correct features from different images and compares them to classify the image. The images are standardized to the size of 224*224*3 (height*width*color channels). The square images can just be zoomed in or out, but the rectangular images are cropped to a square of shorter length and then adjusted to the specified size.

The next step is data normalization. Images are 2-Dimensional (2D) matrices with a pixel value in each cell. These pixel values range from 0-255. Performing computations within this range is a lossy effort. To reduce the load on the system, these pixel values can be normalized to a much smaller range, in this case, it is between 0-1. For this, each value is divided by the highest possible value which is 255 and the resultant is stored in the cell. This ensures that each value within a block is between 0-1.

2.2.2. Model definition and training

The proposed CNN model consists of an Input layer of size 224*224*3, which is the image obtained from the dataset. The next phases can be divided into the convolution phase and the neural network phase. The main function of the Convolution phase is to obtain a feature map from the existing image. Figure 2 shows the proposed CNN architecture. The proposed CNN architecture consists of 4 main layers:

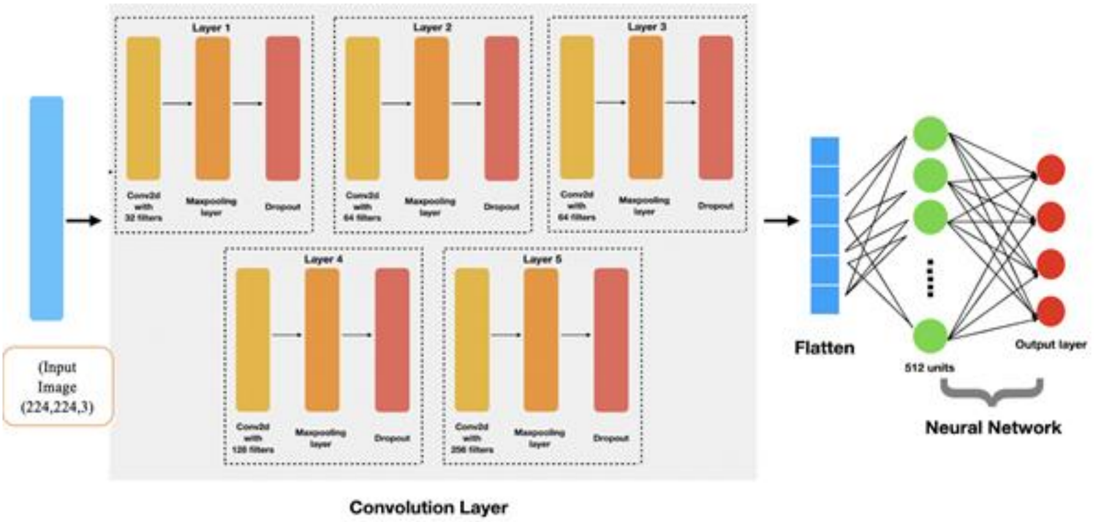

Figure 2. Proposed CNN architecture

- Convolution layer: the convolution layer is used to extract the features from the existing image. It does so by converting the image into an np array and also ensures that the patterns in the image are not lost. It uses a mathematical operation to multiply a small matrix known as a filter with each part of the image.
- Activation layer: activation functions are a fundamental part of neurons [25]. They are mainly found in the hidden layer which helps in identifying complex features based on the number of layers.
- Pooling layer: The pooling layer is a type of summarization and integration layer which replaces the output of certain instances in the matrix. It reduces the amount of effort required by further layers by eliminating unnecessary features and also reduces the size of the image as the process proceeds. The pooling system is done on each filter equivalent to the image and finally summed up to provide to the next layers.
- Fully connected layer: The last part of the CNN network is the Fully connected layer. It is a simple neural network in which the output of every node in the nth layer is connected as an input to the n+1th layer. Before providing the features to the Fully connected layer, the features are flattened into a single dimension for making the analysis process easier. The fully connected layer then analyses these features and tries to identify the class of the input image.

There are 5 convolution layers, each of which consists of conv2d with filters increasing from 32, 64, 64, 128, and 256 respectively. After the Conv2d is the max pooling layer which uses a 2×2 filter for reducing the dimensionality of the image and preserving the dominant features. Finally, in each layer, a proportion of the neurons are left in the inactive state to prevent the model from encountering the overfitting problem. The activation function used in each of the convolution layers is rectified linear unit (ReLU). ReLU sums up the outputs from each neuron and applies the function of treating the values below 0 as 0 and the values above 0 are left unchanged.

After the 5th layer of the convolutional phase, the output matrices are flattened into a single dimension for providing input to the neural network phase of the model. The flattened cells are treated as input to the neural network phase which consists of a single hidden layer with 512 cells. These cells analyze the various features and try to match them with the ones already labeled. The probability for the image to belong to any of the four classes is given as output finally. This is shown in the last output layer containing 4 neurons. The output at each node is the probability for the image to belong to its class. The SoftMax function is used in the end to scale the outputs into probabilities.

The output nodes contain some values. The function of SoftMax is to scale these outputs to probabilities. The total sum of probabilities is 1. The model is trained for 100 epochs and in each epoch, a batch of 32 images was provided each time to train the model.

2.2.3. Model evaluation

Model training and model evaluation complement each other. The model is trained for a given number of epochs. After each epoch, the model is compared with the previous best model and if the present model is performing better than the current best model then the current best model is modified to the present best model. In each iteration, the image is forwarded from the input state to the feature state and finally to the neural network model which classifies the image. This whole process is called a forward pass. The single movement from the initial to the final state is called an iteration. Updating the weights is what essentially meant by model learning. We then instruct the model with a list of parameters to train the model. These include the entire number of iterations and the appropriate loss function applicable for the model. Loss functions are used to retrain the model by backpropagating its mistakes. An epoch consists of several iterations. Parameters that help to fine-tune the model architecture are called hyperparameters and the sequence of identifying the best model is called tuning of hyperparameters. Validation data helps to perform this tuning process. In order to choose the best parameters for each layer, a technique called RandomSearch was used. In random search, the possibilities of parameter values are stored as a list and all the possibilities are tried for each model and the best model is finally provided. This provides a foolproof method for deciding the number of parameters. It also ensures that the best model is selected of all the possibilities. The criteria for deciding the best performing model is the 'validation loss'. The reason for selecting validation loss instead of training loss is that the performance can be better analyzed by checking it on data other than the training data. It shows how generalized the model is. So, the loss on the validation data is taken as the criteria for deciding the best model. The model with the least validation loss is taken as the best model. Since we require fewer misclassifications rather than better accuracy, the loss was taken as the criteria for selecting the best model. The loss and accuracy growth for each epoch can also be visualized for the best model. This presents the overall view of the model and can help to debug the anomalies.

2.2.4. Graphical user interface

A project with the best model can fail to impress if there is no medium to showcase its performance. Therefore, the creation of a graphical user interface (GUI) for the proposed system is the most important step in the development. Apart from developing an accurate CNN model for the purpose of disaster identification, providing an interface in an appealing and aesthetic way for using it is equally critical. The proposed model is to provide an Interface to predict the Disaster and display it to the user. Figure 3 shows the proposed layout for the user interface and the process of interacting with it.

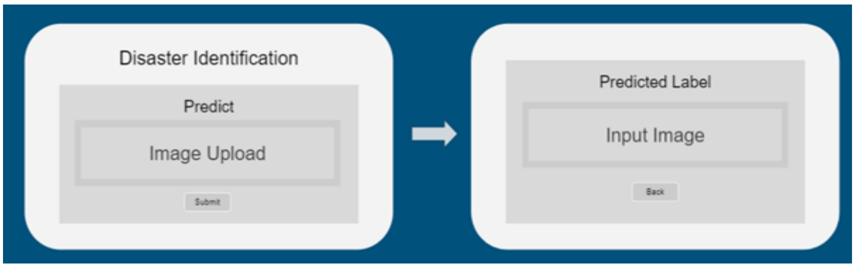

Figure 3. GUI layout

The first step in this GUI making is the homepage. The user interface should attract the user and has to be neat, clean, and simple. So, this model uses the glass morphism technique which is the trending technique that fulfills all the above needs. Essentially, the main aspect of this trend is a semi-transparent background, with a sublime shadow and border. The effect is based on background blur with transparency and uses stacked layers to show the depth and context of the interface. The Home page consists of the name of the system and a dropbox to upload an image or even to drag or drop the image, where the user can upload the image and submit it for the prediction. The next step is connecting the front end to the model for the prediction of the output. The model makes extensive use of python libraries. Integrating python into HTML-Hyper text mark-up language and cascading style sheets (CSS) is a difficult task. For this, we make use of Flask. Flask provides support for python in web pages and is lightweight and can be used on any platform. The best model is integrated into the deployment process. The model is linked to the submit action of the Interface. Once the user clicks the submit button, the image is given to the model in the background and the entire process runs inside. Finally, the predicted label is returned as output for the image. This predicted label is displayed on the webpage along with the image.

3. RESULTS AND DISCUSSION

The final model was obtained and its performance was checked over each epoch. The accuracy seems to be gradually increasing up to 0.92 and the loss was decreasing to 0.17. The model accuracy graph and model loss graph are shown in Figures 4.

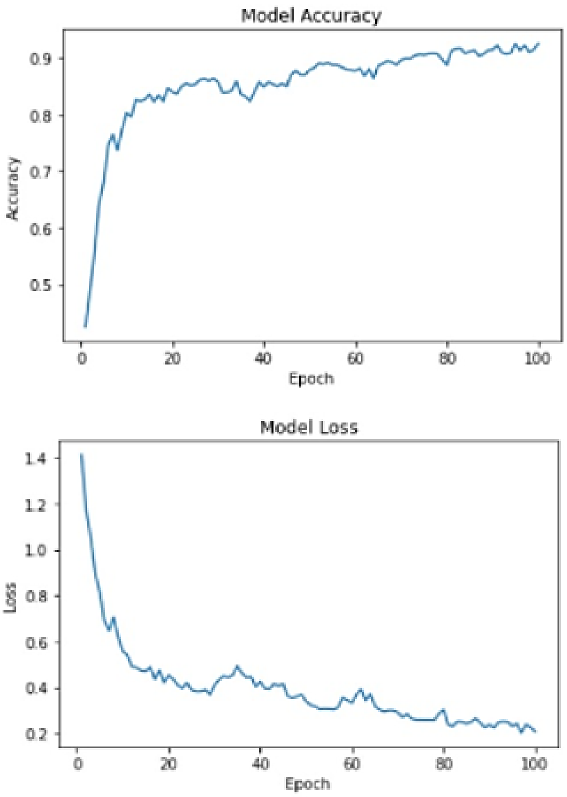

Figure 4. Model accuracy graph

This shows that the model has been learning new features over each epoch and in turn using the learned features to predict the result for foreign images. The increasing trend of accuracy shows that the model does not face any issues with the data. The training images seem to be distinct enough to train the model on all possibilities of the orientation of the data. The model accuracy kept continuously increasing as the epoch progressed. The model loss is continuously decreasing over each epoch. Initially, the loss is high due to the random assignment of neuron weights. The loss after each epoch is backpropagated and weights are changed accordingly. This constant learning over epochs helps to reduce the loss. The number of misclassifications also reduces over each epoch. In each epoch, some mistake is identified and is rectified. The final model was thus the model with the best accuracy and the least loss. To check that the model is free from overfitting, the model was tested with a sample test set of 400 images. These images were never given to the model. The model predicted the outputs of these images and the accuracy was 91% which was close to the train set. The python imaging library (PIL) library was included to show images and the outputs generated from the trained model were displayed as a plot. 400 images from the test set were used to perform predictions and the results were convincing. A confusion matrix was drawn for the classifications on a set of test images. Figure 5 shows the confusion matrix for the predictions.

It can be seen that the diagonal elements are more in number. These are the number of correct classifications. The scarce values spread in the other cells are the number of misclassifications. We can see that the model detects cyclones and earthquakes better than floods and wildfires. The ambiguity sometimes occurs when the image belongs to a flood and is classified as an earthquake due to its wreckage. Also, the images of wildfires are sometimes misclassified as cyclones due to the similarity of the smoke color of wildfire and the watercolor in cyclones. The performance of the model was also visualized by calculating the metrics such as accuracy, f1-score, and support. The model tends to work better if the accuracy is higher. The model designed has an accuracy of 91%. An interactive user Interface was created for the user to upload an image. The home page allows the user to upload an image and click on the submit button. Another remove option was also present at the top right corner if the user wants to change the image. The predicted class is shown on the next webpage. Figure 6 shows the Graphical User Interface for the designed system.

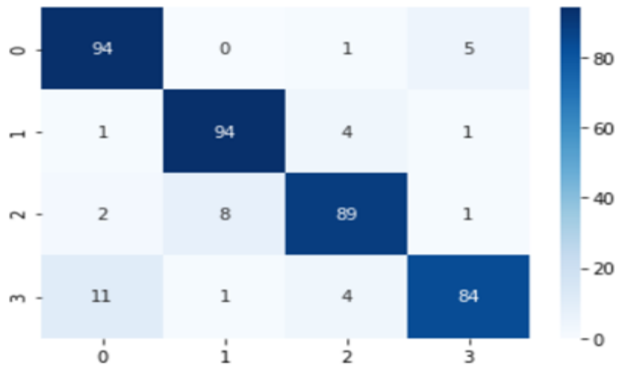

Figure 5. Confusion matrix

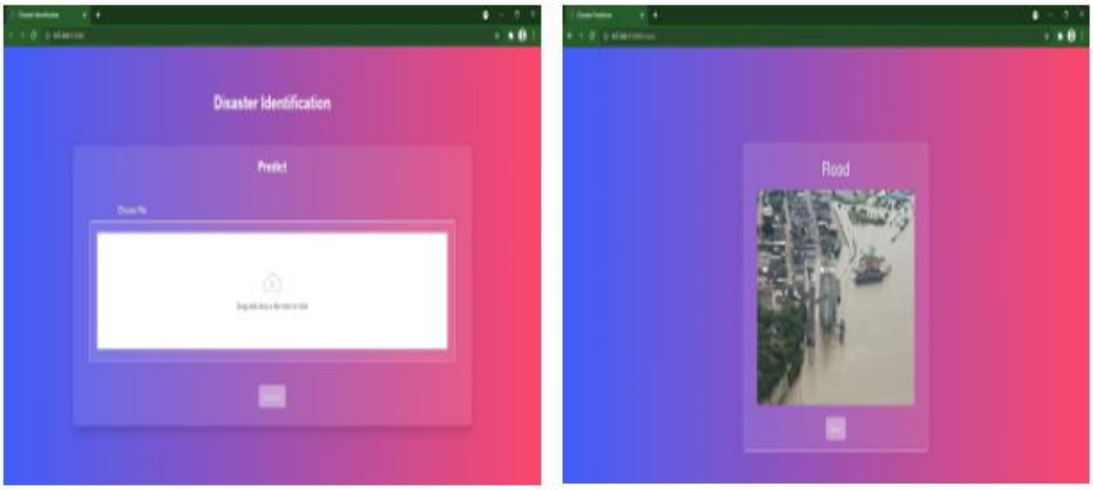

Figure 6. Graphical user interface

4. CONCLUSION

The proposed system provided convincing results with an accuracy of 92%. The 9,600 images used for training the model proved to be better than the original un-augmented dataset. The accuracy of the model with the original dataset was 88%. The significant improvement in the model is due to the extra orientations provided due to augmentation. The model was able to capture the features of the images provided and the neural network was able to use these features to classify the images correctly. The current system includes the identification of four disasters-cyclones, earthquakes, floods, and wildfires. Flask API was used to integrate the CNN model to a webpage and test it on other images. This work can be extended to include other disasters and the model can also be improved further by appending new images to the existing dataset. The proposed system can further be extended to identify the disasters from video footage. This can help in live detection of the disaster from the camera footage.

REFERENCES

[1] M. Tonini, M. D'Andrea, G. Biondi, S. Degli Esposti, A. Trucchia, and P. Fiorucci, "A machine learning-based approach for wildfire susceptibility mapping. The case study of the liguria region in Italy," *Geosciences*, vol. 10, no. 3, p. 105, Mar. 2020, doi: 10.3390/geosciences10030105.

[2] A. R. M. T. Islam *et al*., "Flood susceptibility modelling using advanced ensemble machine learning models," *Geoscience Frontiers*, vol. 12, no. 3, p. 101075, May 2021, doi: 10.1016/j.gsf.2020.09.006.

[3] C. Tang, Q. Zhu, W. Wu, W. Huang, C. Hong, and X. Niu, "PLANET: Improved convolutional neural networks with image enhancement for image classification," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–10, Mar. 2020, doi: 10.1155/2020/1245924.

[4] Li, Zhao, Zhang, and Hu, "Detection of wildfire smoke images based on a Densely Dilated convolutional network," *Electronics*, vol. 8, no. 10, p. 1131, Oct. 2019, doi: 10.3390/electronics8101131.

[5] S. Mangalathu and H. V. Burton, "Deep learning-based classification of earthquake-impacted buildings using textual damage descriptions," *International Journal of Disaster Risk Reduction*, vol. 36, 2019, doi: 10.1016/j.ijdrr.2019.101111.

[6] M. Aamir *et al*., "Natural disasters intensity analysis and classification based on multispectral images using multi-layered deep convolutional neural network," *Sensors*, vol. 21, no. 8, p. 2648, Apr. 2021, doi: 10.3390/s21082648.

[7] S. Panigrahi, A. Nanda, and T. Swarnkar, "Deep learning approach for image classification," in *2018 2nd International Conference on Data Science and Business Analytics (ICDSBA)*, Sep. 2018, pp. 511–516, doi: 10.1109/ICDSBA.2018.00101.

[8] H. S. Obaid, S. A. Dheyab, and S. S. Sabry, "The impact of data pre-processing techniques and dimensionality reduction on the accuracy of machine learning," in *2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON)*, Mar. 2019, pp. 279–283, doi: 10.1109/IEMECONX.2019.8877011.

[9] G. G. Patil and R. K. Banyal, "Techniques of deep learning for image recognition," in *2019 IEEE 5th International Conference for Convergence in Technology, I2CT 2019*, 2019, doi: 10.1109/I2CT45611.2019.9033628.

[10] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017, vol. 2018-Janua, pp. 1–6, doi: 10.1109/ICEngTechnol.2017.8308186.

[11] R. Chauhan, K. K. Ghanshala, and R. Joshi, "Convolutional neural network (CNN) for image detection and recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Dec. 2018, pp. 278–282, doi: 10.1109/ICSCCC.2018.8703316.

[12] N. Jmour, S. Zayen, and A. Abdelkrim, "Convolutional neural networks for image classification," in *2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, Mar. 2018, pp. 397–402, doi: 10.1109/ASET.2018.8379889.

[13] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[14] E. Chang, M. T. Wang, R. Chen, S. P. Tan, and S. Y. Shen, "Design trend of graphical user interface," in *2014 International Symposium on Computer, Consumer and Control*, Jun. 2014, pp. 962–965, doi: 10.1109/IS3C.2014.252.

[15] P. Vogel, T. Klooster, V. Andrikopoulos, and M. Lungu, "A low-effort analytics platform for visualizing evolving flask-based python web services," in *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, Sep. 2017, vol. 2017-Octob, pp. 109–113, doi: 10.1109/VISSOFT.2017.13.

[16] Y. Liu, Y. Zhou, S. Wen, and C. Tang, "A strategy on selecting performance metrics for classifier evaluation," *International Journal of Mobile Computing and Multimedia Communications*, vol. 6, no. 4, pp. 20–35, Oct. 2014, doi: 10.4018/IJMCMC.2014100102.

[17] X. Jiang, Y. Wang, W. Liu, S. Li, and J. Liu, "CapsNet, CNN, FCN: Comparative performance evaluation for image classification," *International Journal of Machine Learning and Computing*, vol. 9, no. 6, pp. 840–848, Dec. 2019, doi: 10.18178/ijmlc.2019.9.6.881.

[18] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of convolutional neural network- a deep learning approach," *Procedia Computer Science*, vol. 132, pp. 679–688, 2018, doi: 10.1016/j.procs.2018.05.069.

[19] C. Khosla and B. S. Saini, "Enhancing performance of deep learning models with different data augmentation techniques: a survey," in *2020 International Conference on Intelligent Engineering and Management (ICIEM)*, Jun. 2020, pp. 79–85, doi: 10.1109/ICIEM48762.2020.9160048.

[20] A. Mikolajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, May 2018, pp. 117–122, doi: 10.1109/IIPHDW.2018.8388338.

[21] J. J. Clague, F. Comiti, T. Glade, J. Goff, and H. K. Gupta, *Natural Hazards*. Springer, 2008.

[22] S. Tripathi and R. Kumar, "Image Classification using small convolutional neural network," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Jan. 2019, pp. 483–487, doi: 10.1109/CONFLUENCE.2019.8776982.

[23] Z. Wang, M. T. Tham, and A. Julian Morris, "Multilayer feedforward neural networks: A canonical form approximation of nonlinearity," *International Journal of Control*, vol. 56, no. 3, pp. 655–672, Sep. 1992, doi: 10.1080/00207179208934333.

[24] A. Mignan and M. Broccardo, “Neural network applications in earthquake prediction (1994-2019): Meta-analytic and statistical insights on their limitations,” *Seismological Research Letters*, vol. 91, no. 4, pp. 2330–2342, Jul. 2020, doi: 10.1785/0220200021.
[25] J. Ding, X. Li, X. Kang, and V. N. Gudivada, “A case study of the augmentation and evaluation of training data for deep learning,” *Journal of Data and Information Quality*, vol. 11, no. 4, pp. 1–22, Sep. 2019, doi: 10.1145/3317573.

BIOGRAPHIES OF AUTHORS

Dr. Mummaneni Sobhana is currently working as Sr. Assistant Professor, Department of Computer Science and Engineering, V R Siddhartha Engineering College, Vijayawada. She received Ph.D. degree in Computer Science and Engineering in 2018 from Krishna University.She has 14 years of teaching experience. Her research interests lie in areas such as Artificial Intelligence, Machine Learning, Data Analytics, Cyber Security and Software Engineering. She published 17 papers in National and Inter- national journals and also published 3 patents. She can be contacted at email: sobhana@vrsiddhartha.ac.in.

Dr. Smitha Chowdary Ch is currently working as Associate Professor, Department of CSE, Malla Reddy Engineering College for Women, Dullapally, Bhadurpally, Kompally, Secunderabad, Telengana, India. Fields of interest are Software Engineering, Internet of Things, Databases, Data Mining, Artificial Intelligence, Machine learning, Data Science.smithacsc@gmail.com

Dr. Devaganugula N. V. S. L. S. Indira is currently working as Associate Professor in the Dept. of Information Technology, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru, AP, India. She completed B.Tech. (CSE) in 2006, and M.Tech. (CSE) in 2009 from JNTUK, Kakinada. Received Ph.D. from Krishna University, Machilipatnam in 2018. Her research interest covers Sentiment Analysis, Data Analytics, Cyber Security, Artificial NN, WSN and Image/Vudio Processing. She Published 41 Papers in Natioanl/ International Journals and presented 14 papers in International Conferences. She has 17 years of Teaching Experience. She can be contacted at email: indiragamini@gmail.com.

Konduru Kranthi Kumar Assistant Professor Department of Compueter Science and Engineering at Dhanekula Institute of Engineering and Technology, with qualification B. Tech, M. Tech, (Ph. D) and having 10 Years of Teaching experience, his research interests are IoT, AI, ML and DL, he has published 33 papers and presented papers in international IEEE Conferences. He can be contacted at email: kk97976@gmail.com.