SMART FLOOD DETECTOR

Authors:

Deeya Gupta-18ucc072@lnmiit.ac.in

Latika Swarnkar-18ucc078@lnmiit.ac.in

Abstract

Floods are one of the most dangerous of all the other acts of god.. Many deaths during flood and money loss suffered yearly in this universe for improved response to flood risks. But above all this, we have great opportunities to explore that can help us in such hazardous situations by capturing images from cameras and using wireless sensors which use the Internet of Things for efficiently improving flood management of flood. Through this paper we can get a good review of the **KERAS' pre-trained MobileNet Convolutional deep neural networks(CNN)** and various applications which are used in reducing and monitoring floods. The report highlights some advanced techniques and IoT-based algorithms which uses computer vision that as a result can be used in challenging flood monitoring and controlling that can occur in upcoming times by sensing conditions of nearby localities.The accuracy we have achieved using above approach is over **98%** which is quite satisfactory in predicting the results

1.1 Introduction

Most flood-prone communities (like coastal areas) experience floods regularly due to heavy rainstorm and unexpected rise in sea level. By one research it was found that by the year 2050 repeated floods will be very common for coastal areas of countries like the US, etc., and will regularly continue for more than 30 days in a year because of unexpected rise in sea level. Roadways services are also very badly affected during floods because of jamming issues.
This report provides a method for monitoring the risk of flood built on images by using KERAS' pre-trained MobileNet Convolutional deep neural networks(CNN) and various computer applications.

In earlier days, we used to focus primarily on remote sensing devices like satellites, aircraft for flood prediction and detection. These remote sensing devices used to provide high-level flood information, but they were insufficient in giving local details such as extent , extremity and floodwater depth in a specific locality.
Various audios, videos and scrutiny methods have also been studied so as to detect change in water level, but in all these methods ,the main problem was water segmentation.

1.1.1 Scope of this Project

This project is used for calculating the risk of flood in an image. This Smart Flood Detector gives authorities an alert(through Twitter) when there is a risk of flood in nearby localities and keeps updating the same through capturing images at regular intervals. Thus it will predict the risk of flood in an area. If there's a chance of flood, the authorities will be informed, and easier evacuation plans and rescue of the people can be made possible.

1.2 Literature Survey

In today's world there are many IoT alterations under development in the area of misadventure management. Currently, there are many systems to monitor the risk of flood in flood-prone localities. Most of the existing systems use sensing techniques and satellite imagery methods which does not guarantee accurate results.Below table shows some of the innovations in this domain:

Name	Technology	Disadvantages	Description
Real-time flood monitoring and warning system	GPRS, VirtualCOM, WAP, Wireless Sensor	GPRS may be unstable and Network coverage area cannot be implemented in this.	In this flood condition is monitored remotely by utilizing wireless sensors network which uses the GPRS communication.
Flood Monitoring using Computer Vision	Sensors, GPRS	Detection algorithm ay give error and it may not send warning messages	In this water level is estimated in a flooded region using the human height as reference.
Urban Flood Monitoring Using Computer Vision	Participatory sensing techniques	No warning system used and hence not Accuracy	This method involves capturing and uploading images of the partially submerged static structures and then estimating flood line against the reference image
Development of a Remote Station for Real-time Monitoring of Urban Flooding	GPRS, Web-based tool, Pressure Sensor	No accuracy, and reliablity	In this an urban flood monitoring station is developed which utilizes pressure sensor to determine flooding levels.

1.3 Techniques Used

1.3.1 Convolutional Neural Network (CNN)

CNN (Convolutional neural networks) is used where there is a requirement of computer vision work. We have used a pre-trained model trained on enormous datasets with massive no of entries like ImageNet, COCO, and many more. Then we have oriented these architectures to work for our unique(flood) dataset ideally, and this process is generally termed transfer learning.

We basically used MobileNet for the beginning part of CNN. This pretrained model is precisely adjusted and trained with the help of a training set and then tested on a testing set.

We worked on Keras and Tensorflow deep learning libraries with Python version 3.8.6. This model is trained with batch size and epochs of size 10.

1.3.2 Mobilenets

MobileNets is a type of deep convolutional neural networks that are very small in size,low-latency,low-power models, and very fast in performance than other models. These models can be used for classification,detection and other common tasks for which CNN is used. Mobilenet model is trained on 1000 different classes, and it has 1000 nodes in its output layer.
In terms of size, the largest MobileNets is 17 MB, while the largest VGG16 model is 533 MB. When we want to deploy our model in the browser then this large size can be a difficult issue to handle. On ImageNet Classification, MobileNets show well-built performance as compared to other preferred models. These models are used in many applications such as large scale geo-localization and fine-grain classification,etc.

1.3.3 Keras

It is developed using open-source machine libraries, like TensorFlow, Theano etc. TensorFlow is the other symbolic math library used for making many neural networks and deep learning models. TensorFlow is easy to work with and is very flexible , its benefit is in distributed computing.

Keras is based on a minimum structure that provides a very easy way for developing various deep learning models which uses TensorFlow or Theano.

Keras was created to define deep learning models and moreover it is the most excellent choice for most of the deep learning applications.

Keras holds many techniques to optimize and make high level neural network API easy to understand and more performing.

Some features includes:-

- Simple and extensible application programming interface.
- Supports several hardware platforms and several backends.
- Has a user friendly framework and can run both on CPU and GPU.
- Highly adjustability of computation.

1.3.4 Tensorflow

It is a framework designed so as to execute various machine learning ideas in easy to learn and code manner. Includes computational algebra methods for calculation of mathematical and analytical expressions.

Because of Tensorflow, google can now easily launch many new ideas and applications and thus tensorflow is also widely known as a google product and can be used in making engines that run **Google** much more powerful than earlier. TensorFlow can be used to test,train and run deep learning neural networks for multiple purposes like image processing and image recognition, handwritten digit classification,and moreover can be used in creation of various sequence models.

Some features includes:-

- Mainly includes deep neural networks and other ML techniques.
- Have high scalable features which can be used for various calculations and computations in data sets.
- Uses GPU for calculations and also includes some great characteristics of optimizing memory.

1.3.4.1 Why is TensorFlow So Popular?

TensorFlow is a well-organised and well-documented open source platform which Includes collection of various libraries in machine learning. Moreover it also includes many important functions and methods.

1.3.5 Flask

It is an Application Programming Interface(API) or web application framework of Python that allows us to create web-applications.
Easy to understand and learn and can be used to build complex web-Application. A Web Framework is basically the collection of interrelated modules and libraries that helps the existing developer to write and code various applications without much bothering about the low-level codes like thread management, protocols etc.
Based on Jinja Python template and WSGI toolkit engine which is used in creating HTML and other markup formats

1.3.5 How is it an IoT project?

An IoT project must have three components: Things, Users, and Applications. Here, we have assumed that the images will be taken with the help of Raspberry PI, and then prediction on those images with our deep learning model will be made. So, Raspberry PI is assumed to be hardware in this system. Then after an image is taken, this image will be tested for prediction on our web app. And after that, if a flood is detected, an alert will be sent to authorities through Twitter. So this alert system which involves user interaction+application+things, makes it an IoT project.

1.4 Methodology

Flood destroys lives and valuable resources every year throughout the world. In this situation, a camera-based solution can be implemented to provide additional metrics such as the severity and location of flood or improve authorities and ordinary people's response times. If the location and nature of the flood are identified, then an automated intervention may be possible. In this way, we can accurately predict the impact of floods and send alerts(through twitter or email) to authorities for rescue. To take images, we can use Raspberry pi and then process our results through our model.
This project aims at developing a flash flood alert system so that further flooding does not cause any harm to people living in that area.
We employ KERAS's pre-trained MobileNet Convolutional deep neural networks(CNN) and fine-tune it to predict the probability of a flood in a room.

- **Obtaining and Organizing the Dataset:-**

  This dataset has been taken from:
  https://drive.google.com/file/d/1oT07-wVuYj_hByn4FqUtIj63Yg_fe7dP/view?usp=sharing
  This dataset has a total of 473 images, out of which 253 are flooding images and 220 are images with no flood(random images). We first of all preprocessed all the images with a target size of 224x224.These images have all the scenes from natural locations,suburban and urban regions.All the images are with other conditions and characteristics.
  These dataset has been divided into train, test and valid sets out of which train and valid sets are used to train and validate our CNN model and test sets are used to test the final model to determine the model's accuracy.

  Following are the sample images from the dataset:

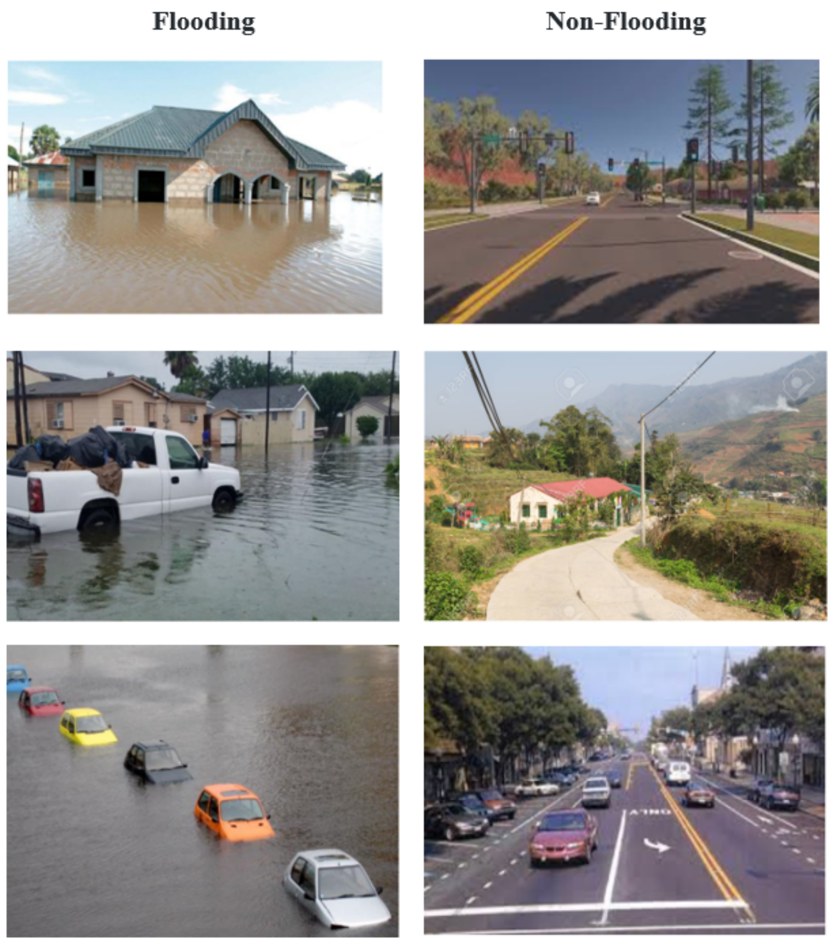


Fig1.4.1: Sample images from dataset

- **Preprocessing and Image Classification**
  We preprocessed all our images using the MobileNet's preprocess_input function with a target size of 224 x 224.
  Then loaded the pre-trained MobileNet image classifier and fine-tuned it to train the model with train and valid batches. This is how we developed an image classification model to categorise different images into flooding and non-flooding.

- **Evaluation and Prediction**
  First, we preprocess our image for evaluation and prediction and then predict using test batches. Then we create a confusion matrix to visualize our predictions and obtain the accuracy.

- **Deployment and Twitter Alert System**
  We deployed our model using the python flask library and ran our model on the website. When we choose an image file, we get the predictions on our website. If the image is a flood image, then a tweet and the flood image are posted on Twitter to notify the authorities for the rescue of people.

1.5 Components

1.5.1 data and evaluate folders

The data folder contains our dataset and it has 3 folders namely train,test and valid.Each folder further contains flooding and normal folder. In the flooding folder, we have all the flooding images and in the normal folder all the normal random images are included.This images are used to train,test and validate our model.
The evaluate folder contains 7 images of both flooding and no flooding type.Using these images, we test or model.

1.5.2 Flood_Detector.ipynb

This is the main file where code for our trained model and code for twitter alert system resides.This code helps us in using our model for testing and other deployment purposes. We are defining two labels in this which are, (1) Flooding andThen we loaded data and preprocessed images according to mobilenet requirements and then created three different batches of the data.
Then we loaded a pre-trained lightweight (2) No Flooding.First we loaded all the required libraries and functions.
mobilenet image classifier and got the summary.
All the layers of the original mobilenet except the last 5 layers are stored in variable x so as to get optimal results for this task.
Last 5 layers of the mobilenet are stored so as to train our model during fine-tuning so as to keep all of the previously learned weights.
Then we created a binary output layer so that our model can be used for binary classification like flooding and no flooding.
Then we complied and trained our model and saved our trained model for future use.
We also tried to make some good predictions and also plotted confusion matrix so as to estimate how well our model is performing.

1.5.3 flood_detection_model.h5

In the .ipynb file when we save our model with the **model.save (" flood _ detection _ model.h5 ")** function then this .h5 file is created. It will include the model's configuration/architecture,compilation info of model ,weights and states of different layers.We can use it for testing and deployment of our model can be done in app.py python script.

1.5.4 For Deployment

1.5.4.1 app.py

In this python script, we have used the flask python library to deploy our model. This is where the code for our web service is developed.By loading the Flood_detection_model.h5 file, we are able to load and use our model for predictions.We have also added the functionality of posting a tweet in this same file.

1.5.4.2 index1.html

In this html file, we have built the front end of our model for displaying the results.Results will be calculated from app.py file and then will be displayed on our webpage.

.

1.6 Implementations

Python 3.8.6 is used as a platform for this project. The deployment code has been written in Microsoft Visual Studio C++ 2015 and IDE for coding our model in Jupyter Notebook.

The code implementation steps are the following:

- We loaded all the python libraries like TensorFlow, Keras, NumPy, Matplotlib, Scikit Learn,etc.

- Then we loaded our data using the image generator() function and passing the required parameters for preprocessing all the images according to the requirements of Mobilenet.Two labels: (1)**Flooding** and (2) **No Flooding** for classification of images.

- Loaded Mobilenet pre-trained the classifier, assigned it to a mobile variable and stored all the layers except the last 5(by doing experiments) to get optimal results. Then added an output layer with two nodes for two classes, **flooding and non-flooding**. After doing all this, we fine-tuned it for our outputs and assigned this sequential model to the model variable.

- After compiling and fitting this model using train and valid batches, we save the model for the future to load it again whenever we want to use it.

- We then test this model using the test batch and plot confusion matrix and get the accuracy,f1-score, and precision score.

- After this, we import the tweepy python library to use Twitter API, and after providing vital info like API key, access token key, etc., we can use Twitter for posting a tweet. Whenever there is a flood detected, a tweet and the image will be posted on Twitter.

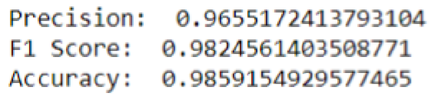
- Then we load our model and deploy it using the Flask Python library. We use an app.py file for this purpose. First, we will be loading many libraries like flask,numpy,tensorflow,Image,etc.Then we create an app variable in which we assign an instance of Flask.After loading the model using load_model() function , we use app_route for creating a new end point. We use POST and GET methods to send our image data and get prediction results.

- For the front end or UI of our web service, we use html, css and javascript in index1.html file. Here we take a response from the app.py file and get the prediction results printed on the screen.

So, this is how our implementation is done.

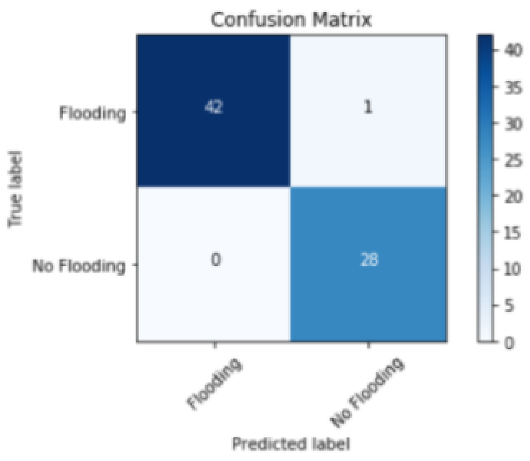
1.7 Results and Analysis

- Our model got both accuracy and an F1 score of around 98% after training for ten epochs. In the confusion matrix plot, we can see that 42 images and 28 images were predicted correctly, and only one was incorrectly predicted.**(Fig1.7.1)**



```
Precision:  0.9655172413793104
F1 Score:  0.9824561403508771
Accuracy:  0.9859154929577465
```

(a)



(b)

Fig1.7.1:(a)Results (b)Plot of Confusion Matrix

- Graph **(Fig1.7.2(a))** shows that the validation and training accuracy becomes almost equal at the end of the 10th epoch, proving that our model is neither over-fitted nor under-fitted.

  Graph **(Fig1.7.2(b))** shows that both training loss and validation loss become almost negligible, indicating correct predictions.

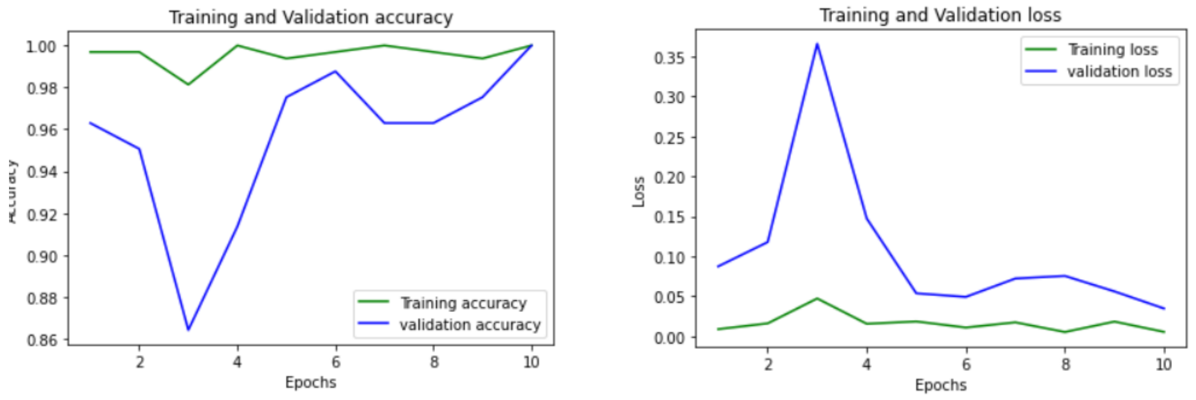


Fig1.7.2:Comparison Plot of (a)Validation and Training Accuracy. (b) Validation and Training Loss

- Fig 3 shows the web app that we created using Flask and HTML/CSS. And when the image is provided to this, then our prediction results appear on the screen.**(Fig1.7.3)**

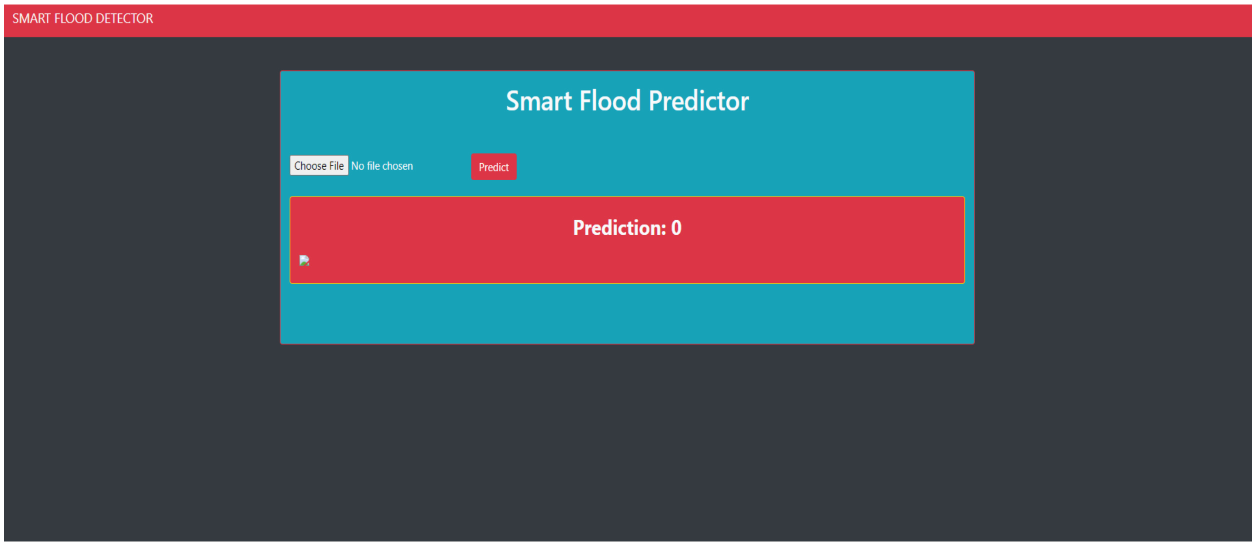


Fig 1.7.3: Web app for Smart Flood Detector

- Fig4 shows that when a 7.jpg image from the evaluate folder is provided, it results in detecting it as a flooding image.**(Fig1.7.4)**

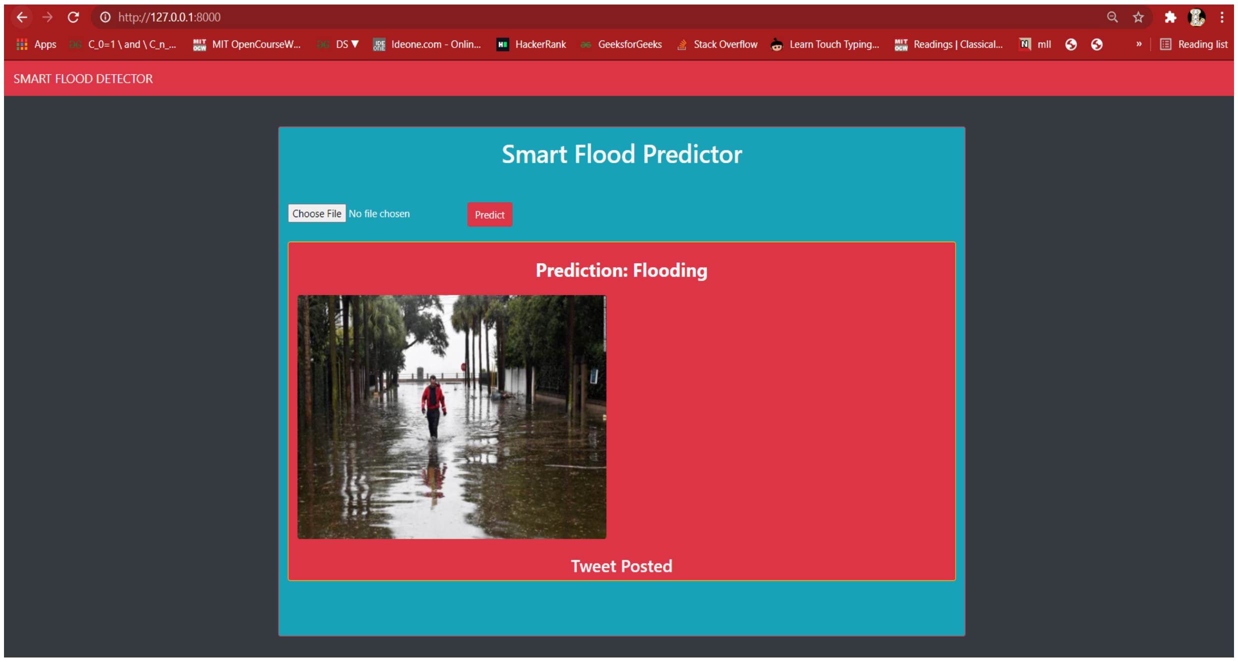
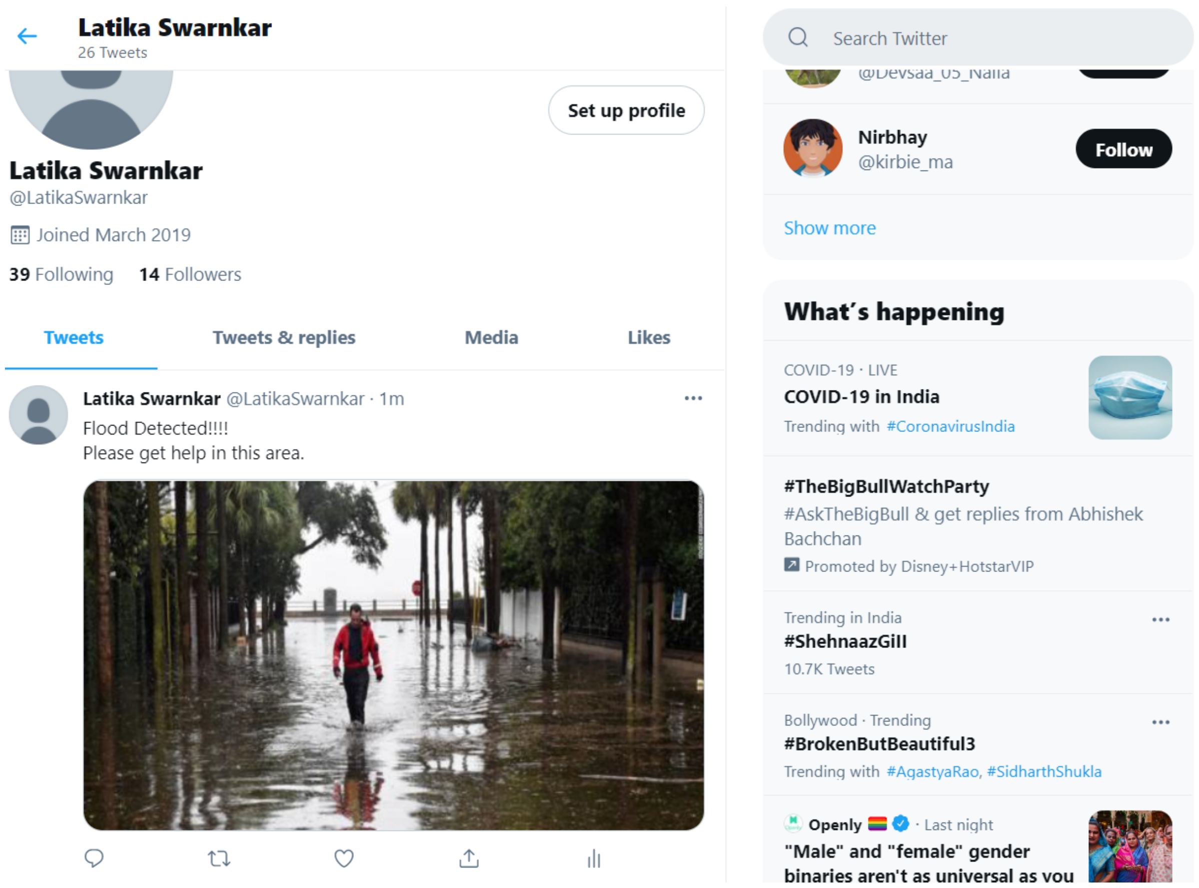


Fig 1.7.4: When a flooding image is chosen

- So, a tweet is also posted at the same time on Twitter indicating a flood in that area. **(Fig1.7.5)**


Latika Swarnkar
26 Tweets

Latika Swarnkar
@LatikaSwarnkar
Joined March 2019
39 Following 14 Followers

Latika Swarnkar @LatikaSwarnkar · 1m
Flood Detected!!!!
Please get help in this area.

Fig 1.7.5: Tweet is posted for flooding image

- But when a 9.jpg file is provided then it predicts it correctly as a non-flooding image and no tweet is posted.(**Fig 1.7.4)**

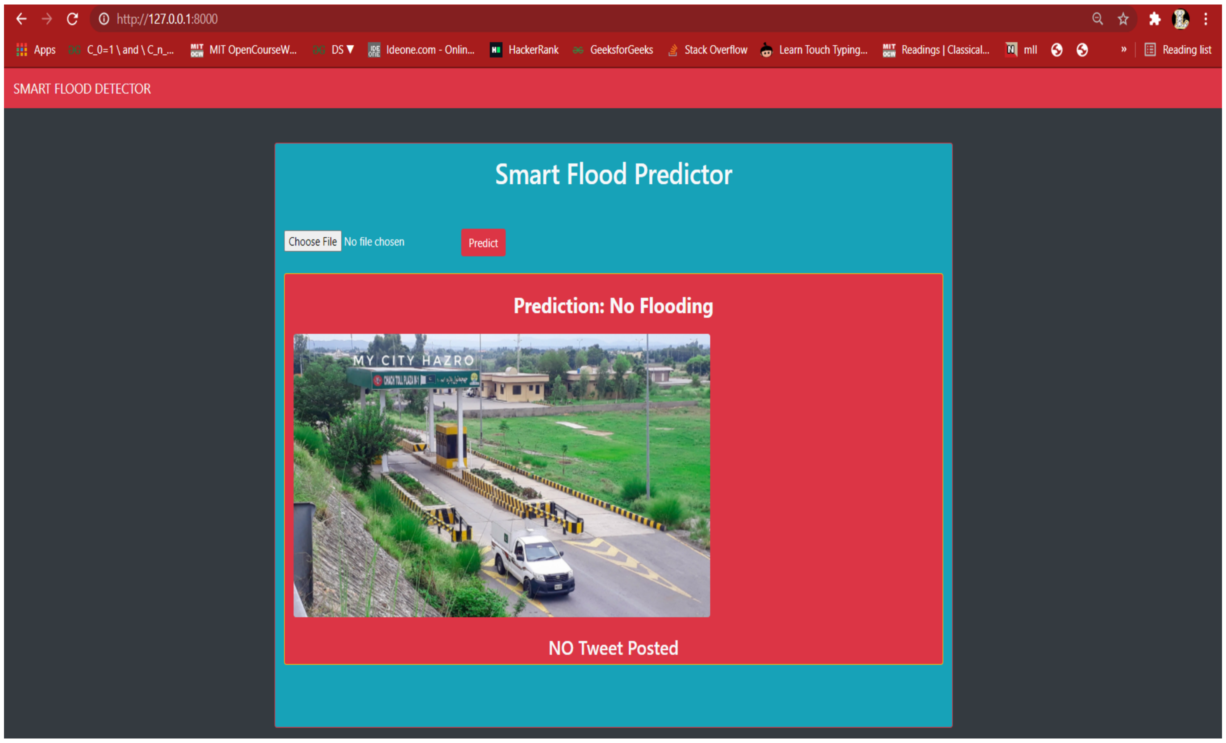


Fig 1.7.4: When a non-flooding image is chosen(no tweet is posted)

1.6 Conclusion

This project provides us with an alert system that will alert us when there is a risk of flood in our locality. It can help our society and humankind with this life-taking natural disaster. Our model has been well tested, and is working correctly, as described in this report. It monitors every aspect that can lead to a flood. It will send a Twitter alert immediately,if the water level rises . An email can also be sent to the responsible authorities if the tweet is not enough .This model can help authorities and government in taking quick and accurate decisions and planning accordingly when there is some mishappening.

Bibliography

[1] https://deeplizard.com
[2] https://arxiv.org/pdf/1909.00125.pdf
[3] https://arxiv.org/pdf/1704.04861.pdf

[4]https://www.pluralsight.com/guides/data-visualization-deep-learning-model-using-matplotlib