Flood prediction based on weather parameters using deep learning

Suresh Sankaranarayanan, Malavika Prabhakar, Sreesta Satish, Prerna Jain, Anjali Ramprasad and Aiswarya Krishnan

ABSTRACT

Today, India is one of the worst flood-affected countries in the world, with the recent disaster in Kerala in August 2018 being a prime example. A good amount of work has been carried out by employing Internet of Things (IoT) and machine learning (ML) techniques in the past for flood occurrence based on rainfall, humidity, temperature, water flow, water level etc. However, the challenge is that no one has attempted the possibility of occurrence of flood based on temperature and rainfall intensity. So accordingly Deep Neural Network has been employed for predicting the occurrence of flood based on temperature and rainfall intensity. In addition, a deep learning model is compared with other machine learning models (support vector machine (SVM), K-nearest neighbor (KNN) and Naïve Bayes) in terms of accuracy and error. The results indicate that the deep neural network can be efficiently used for flood forecasting with highest accuracy based on monsoon parameters only before flood occurrence.

Key words | Bayes, deep neural network, K-nearest neighbor, support vector machine

Suresh Sankaranarayanan (corresponding author)
SRM Institute of Science and Technology,
Chennai,
India
E-mail: *pessuresh@hotmail.com*

Malavika Prabhakar
University of Southern California,
Los Angeles, California,
USA

Sreesta Satish
University College Dublin,
Belfield, Dublin,
Ireland

Prerna Jain
Dalhouise University,
Halifax,
Canada

Anjali Ramprasad
University of Massachusetts Amherst,
Amherst, Massachusetts,
USA

Aiswarya Krishnan
Zomata,
Chennai,
India

INTRODUCTION

Extreme climatic environments and altering weather are the consequences of global warming which have significantly increased the frequency and severity of floods. Floods are one of the most commonly occurring disasters and have caused significant damage to life, including agriculture and economy. They are usually caused in areas where there is excessive downpour and poor drainage systems.

Nowadays, with the growing age of the internet and connectivity among people, information sharing has become more convenient.

While some districts are worse affected in comparison to others, the deployment of an early warning system always prevents excessive damage. In the past few years, many flood risk technologies have been advanced to minimize the after-effects of a flood. In developing countries, this is an important technology where flooding leads to massive loss of life and property. With the introduction of flood monitoring systems (Adam & Pagano 2016), devices will alert before the onset of significant water damage. These systems deliver an early cautionary and precise collection of data during risky weather environments globally. Figure 1 shows the Flood Monitoring and Warning System.

As disasters lead to a huge amount of losses, Internet based technology could aid in monitoring floods (Mitra *et al.* 2016). As the Internet is being used in every common man's life, most of the information is either forecast or an alert is communicated to smart devices. These include

doi: 10.2166/wcc.2019.321

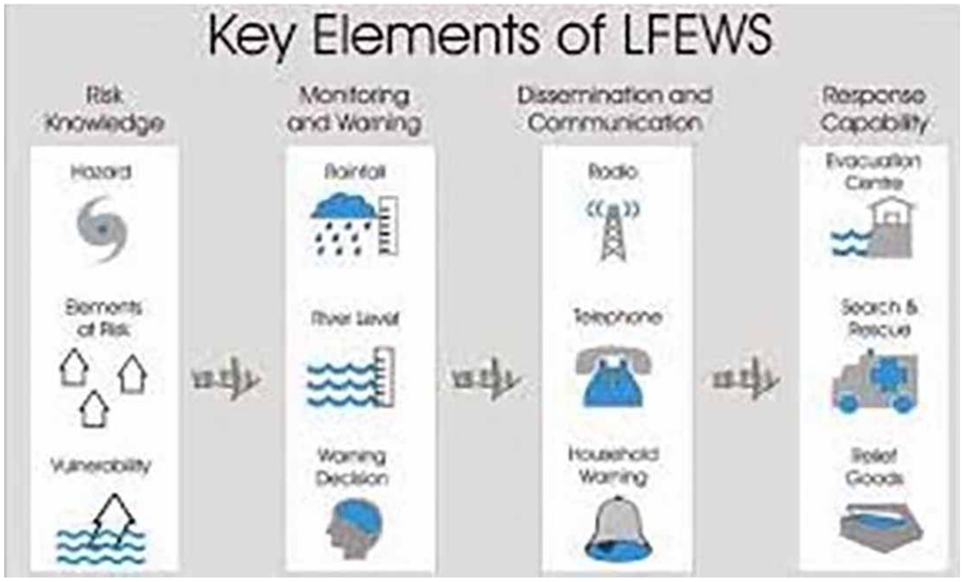


Figure 1 | Overview of Flood Monitoring and Warning System (FLEWS n.d).

various sensors such as Global Positioning System (GPS), Global Navigation Satellite System (GNSS), ultrasonic sensors which are rapidly deployable, always mobile, and provide very precise and accurate locations (Han & Coulibaly 2017). All vital information can be easily shared among people due to growing connectivity. In addition to collecting data for measuring the level of water and defining the scope of a flooded area, radar imagery and optical imagery remote sensing technologies are also widely used.

In one of the studies mentioned here (Chuntian & Chau 2002), a conflict decision model has been developed for reservoir flood control by employing a three person multi objective. The model is very simple and adaptable to real world problems. This has been applied to Fengman Reservoir in China.

Currently, the Internet of Things (IoT) is being widely used in many fields of research, one of them being flood management. One such technology is wireless sensor networks (WSN). These devices mainly consist of spatially dispersed sensors for tracking and recording the physical conditions of an environment. One such technology towards flood disaster prediction is the combination of IoT and neural networks.

Researchers have worked towards developing an IoT-based approach that uses a smart lattice-linked WSN (Mitra *et al.* 2016). The algorithm for prediction used is a meek artificial neural network (ANN) model. The parameters used are precipitation, flow of water, pressure, moisture and the water level is the output variable. WSN simulation results with a higher number of nodes could perform better.

Wireless Sensor network technology is used in water level monitoring and other application areas. However, it is not efficient and the real time notification system is still an issue (Khalaf *et al.* 2015). The three water level sensors placed at dissimilar locations are used to predict the flood and text messages are sent to the flood management center every 30 minutes about the present situation. The data obtained from the hardware is then classified using four algorithms – random forest, decision tree, bagging and hyper pipes, where the first one listed obtained the greatest results with an accuracy of 99.5%.

Bande & Shete (2017) developed an IoT-based flood monitoring system employing an artificial neural network where temperature, pressure, humidity, rainfall and water level data were collected for temporal analysis for flood prediction. The IoT approach is used for data collection from sensors and ANN is used for data prediction.

A flood disaster warning system on Go Research (Abdullahi *et al.* 2018) was carried where NodeMCU ESP8266 was deployed for transmission of sensor data to ThingSpeak Cloud for real time visualization and data storage. A two class neural network model of Microsoft Azure was used for predicting flood status based on a predefined rule. The prediction results proved that the highest accuracy of 98.9% and precision of 100% was achieved with three hidden layers.

In Liu *et al.* (2017), ANN is the main body for flood prediction. Another technology brought into light is the Bayesian Forecasting System (BFS). It offers an ideal framework that takes into account an uncertainty quantification and provides probabilistic flood forecasting through a deterministic model. By also introducing smart drainage systems, severity of urban floods can be minimized. An early warning system that includes the usage of WSNs is one of the solutions that can significantly curtail the challenges brought by floods.

Apart from IoT, the usage of analytics has come up with solutions regarding flood prediction and management. Traditional methods such as image classification, feature selection, signal unmixing and regression were implemented (Khalaf *et al.* 2015). MAP rain is a technology towards providing a flexible and scalable cloud-based solution for flood monitoring analytics. MAP rain technology (MAP Rain n.d) was used for forecasting actual and predicted rainfall for any location. In addition, e-mail alerts were received for higher error values towards forecasting. Similarly, rainfall return periods for any historic, forecast or rain event were calculated.

Flood hazard reduction is another ML-based solution to mitigate the consequences of floods (Camps-Valls 2009). Here, flood analysis includes: evaluating the flood characteristics to choose, implement and operate the measures to prevent and/or mitigate flood impact; understand the hydrological process of formation and dissemination of floods.

Han & Coulibaly (2017) focused on a Bayesian forecasting system (BFS) that takes into account uncertainty quantification and provides probabilistic flood forecasting through a deterministic model. Bayesian methods include 'Probabilistic River stage forecast (PRSF) and Probabilistic Quantitative Precipitation Forecast (PQPF), Bayes Ensemble Forecast (BEF), Probabilistic Flood Forecast (PFF)' provide an efficient flood forecast while accounting for uncertainties.

The recent floods in Chennai and Kerala can be considered as urban floods and considering factors such as alteration in the climate, pressure etc. and the seriousness of urban surges is expected to intensify. If drainage systems are not more intelligent, urban floods will continue to devastate the area.

In other research, two meteorological stations (Rasht and Lahijan, located in Gilan province in northern Iran) over the 2006–2016 time period were used for predicting the evaporation component (Moazenzadeh *et al.* 2018). Pearson correlation coefficient was used for evaporation prediction based on metrological parameters recorded. An SVR model coupled with a firefly algorithm (SVR-FA) was deployed to simulate evaporation values on a daily scale.

Ghorbani *et al.* (2018) used a multi-layer perceptron technique embedded with a Quantum behaved particle Swarm Optimisation algorithm for estimating the rate of evaporation over a daily forecast horizon. This was carried out for the 2012–2014 dataset from Talesh metrological station in northern Iran. The accuracy of this model, which is MLP-QPSO, was compared with other hybrid models – MLP-PSO and a standalone MLP model.

Also, Fotovatikhah *et al.* (2018) carried out a comprehensive survey about the application of computational intelligence (CI)-based methods in flood management systems. Flood debris forecasting surveyed in terms of accuracy and error rate was promising.

In respect of forecasting medium and long-term runoff time series, an artificial neural network (ANN) model was coupled with the ensemble empirical mode decomposition (EEMD) (Wang *et al.* 2015). The two annual runoff time series were analysed for Biuliuhe and Mopanshan in China. From the analysis it is clear that the proposed EEMD-ANN model achieved a significant improvement over ANN.

This research proposes a system known as RAPIDS that stands for Radar Pluvial flooding Identification for Drainage System (Duncan *et al.* 2013). This system aims at predicting the flooding of multiple urban areas using multi-output ANN by considering urban drainage systems and rainfall data particular to the area. The result obtained proved that a single ANN was able to provide an acceptable level of flood prediction but this can be further enhanced by pruning the hidden nodes and analyzing each one. Also, the prediction of most extreme events accurately should be checked by using different performance metrics.

Camps-Valls (2009) focused on traditional and new trends for remote sensing data processing in ML. Traditional methods such as image classification, feature selection, signal unmixing and regression were implemented and new trends include manifold learning, semi-supervised learning, transfer learning, active learning and structured learning.

Hossain *et al.* (2014) proposed a flood warning system that used a neuro-fuzzy technique to centralize the input parameters, such as river water flow, rainfall, water level from a location, which is then determined by the Adaptive Neuro-Fuzzy Inference System (ANFIS) model. Using an adaptive flood warning system, the possibility of flood in an area can be determined.

Liu *et al.* (2017) proposed a deep learning algorithm with Social Adaptive Ensemble (SAE) and BPNN. Using a dataset containing data regarding flow of water in flood years for the period 1998–2010, the first ten years' data were used as training data and the remaining years as testing data. Compared to the traditional models, which showed a problem with large deviations and performance issues, SAE-BP provided accurate prediction results.

Data available from a Geographical Information System (GIS) (Opella & Hernandez 2019) was employed for producing a reliable flood susceptibility and probability map. Fusing Convnet and SVM methods were used for classification and regression analysis. The final output produced from SVM is based on the distinct image prediction output from a dilated convolution and deconvolution network.

Floods are quite commonly occurring disasters in Bangladesh (Ganguly *et al.* 2019). Therefore, in this work, household level flood damage analysis was performed for the 2004–2009 period flood data from 64 districts. Based on the data gathered, machine learning based prediction was employed, namely 'linear regression, random forest and artificial neural network' and compared accordingly. In addition to prediction, principal component analysis was performed towards clustering different variables into predictor groups and inspecting their effect on flood damage.

With the advent of IoT and machine learning algorithms, extensive work has been carried out in deploying sensors and machine learning algorithms such as ANN for predicting flood with an alert system. Many works have focused on the use of an ANN algorithm for predicting the flood based on water level, water flow, temperature, rainfall, humidity and so. Some have applied other machine learning algorithms such as SVM, Bayesian and decision tree. There has also been work that focused on a deep learning-based approach using SAE which predicts the flood based on stream flow.

The major drawback in all these systems is that most flood warning or monitoring systems have focused on employing an ANN with a single hidden layer for predicting the flood based on temperature, rainfall, water flow or so. Some have focused on predicting flood based on water flow level using an SAE based deep neural network. Also, researchers have employed machine learning based prediction, namely linear regression, random forest and ANN, and compared them accordingly. In addition to prediction, principal component analysis was performed towards clustering different variables into predictor groups and inspecting their effect on flood damage.

However, none of the systems relied only on seasonal data, which is rainfall intensity and temperature, for predicting flood at an early stage. Systems so far have also taken water flow and water level into consideration, along with other parameters, in predicting the flood.

Considering all these systems, we have been motivated to develop a deep learning-based flood early warning system for India where temperature and rainfall intensity are taken into consideration to predict the possibility of flood occurrence before there is heavy stream flow and water levels leading to flood. This would be compared with the traditional machine learning algorithms such as SVM, Bayes and KNN in terms of error and accuracy for providing the best machine learning based approach for flood prediction at an early stage before flooding occurs. The Flood Early Prediction System is based on the historical data of two Indian cities, Orrisa and Bihar, as a case study for developing the system. The two cities were chosen because Bihar is one of the most flood-prone states in the country, with about 73% of its total surface area getting flooded annually (as quoted by the International Water Management Institute (IWMI)). Orissa also has a very concerning frequency of flood occurrence across its various districts and hence was included in the regions to be further studied. This prediction would enable appropriate measures to be taken and people could be evacuated in a timely manner before having an adverse effect on human lives, property and so on.

The main objectives of the paper are as follows:

- collection of dataset pertaining to temperature, rainfall and flood occurrence and decomposing it as one single dataset;
- validation of SVM, KNN and Bayes for flood prediction;
- validation of deep neural network for flood prediction;
- comparative analysis of SVM, KNN, Bayes and deep neural network for accuracy and error for flood prediction.

The rest of the paper is organized as follows: the Materials and methods section discusses the theoretical background on flood early prediction using deep learning and different machine learning algorithms followed by system architecture. The Results and discussion section discusses different machine learning algorithms for flood early prediction based on a dataset with accuracy and other metrics for comparison followed by the Conclusion and future work section.

MATERIALS AND METHODS

Floods are one of the most commonly occurring disasters and have caused significant damage to life, including agriculture and economy. They are usually caused in areas where there is excessive downpour and poor drainage systems. These systems deliver early cautionary and precise collection of data during risky weather environments globally.

The system we have developed takes only seasonal data (torrential rainfall and temperature alone) into consideration in predicting the possibility of flood occurrence in a region before there is heavy stream flow and a rise in water levels leading to flood.

Accordingly, rainfall intensity and temperature were taken into consideration for predicting flood at the early stage by employing a deep neural network with multiple hidden layers. This would enable the prediction of flood occurrence probability based on current seasonal weather conditions (temperature and rainfall) as it happens based on sensor data. This would ultimately result in obtaining the best accurate prediction for flood warning using a deep neural network when compared with other machine learning classification algorithms such as SVM, Naïve Bayes and KNN. The data provided in the dataset consists of flood resource variables such as precipitation and maximum and minimum temperature for different regions. This research will examine the data available and attempt to predict the possibility of occurrence of a flood in the districts of Bihar and Orissa and issue an early warning.

The flood dataset consists of labeled training data, which can be classified under binary classification. We can determine the best model parameters to predict unknown labels on other objects (data) by fitting the training dataset. This can be achieved by using traditional ML algorithms such as SVM, KNN and Naïve Bayes. In order to achieve a higher accuracy and exploit the already existing algorithms, we aim to implement a deep neural network for flood prediction. The algorithms which were implemented are explained below.

Support vector machine

The support vector machine is mainly associated with learning algorithms that analyzes data which is used for the study of regression and classification. A set of data that has been trained is chosen as an example, each labeled as fitting into a one- or two-type category. An SVM training algorithm basically creates a model that designates examples to one category or the other, developing a binary, non-probabilistic linear classifier (Gereon 2018; Sankaranarayanan & Mookherji 2019). In Figure 2, SVM has been represented with the help of the examples marked as points in space. Here, each example belongs to one category or the other wherein distinguished categories are divided by a clearly visible space or gap. New values are taken into account each time and then mapped into the same space. SVM is used for the prediction of new data and the class they belong to, based on the distance from the spatial line.

A hyperplane or the spatial line demarcates the binary SVM algorithm while classifying (Color Classification of images with Support Vector Machine 2018). Using data for supervised learning (the data used for training is labeled), the algorithm constructs an appropriate hyperplane which in turn classifies new values incorporated. Considering it two-dimensionally, a hyperplane is basically a line splitting

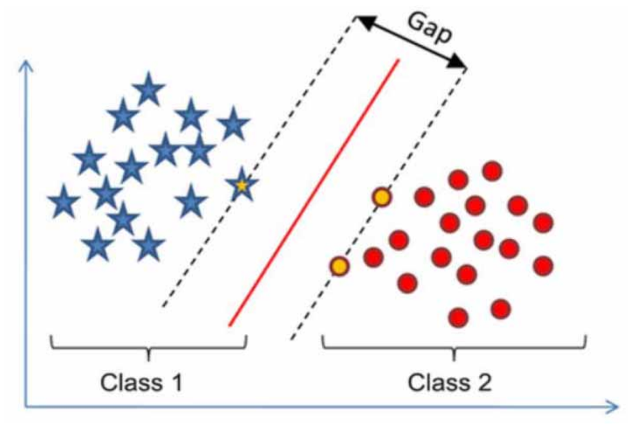


Figure 2 | SVM classification (https://medium.com/deepvision/color-classification-with-support-vector-machine-755360c96037).

the given dataset into halves where individual sides store data from the two previously established classes.

In a support vector machine, a hyperplane is defined as:

$$f(x) = \beta_0 + \beta^T X \tag{1}$$

where β is defined as the weight vector and β_0 as the bias. A hyperplane can be represented in different ways as it is defined by the scaling of β and β_0. Conventionally, a hyperplane can also be defined as:

$$|\beta_0 + \beta^T X| = 1 \tag{2}$$

where the symbol χ represents examples of the closest training values. A canonical hyperplane refers to the hyperplane which contains training values close to the borders. This is mainly seen in support vectors. Here, the distance between χ and a hyperplane β_0 is calculated as:

$$distance = \frac{|\beta_0 + \beta^T x|}{\beta} \tag{3}$$

Considering a canonical hyperplane, the value of the numerator will be equal to one and the distance to the support vectors is calculated as:

$$distance_{support\ vectors} = \frac{\beta_0 + \beta^T X}{\beta} = \frac{1}{\beta} \tag{4}$$

$$M = \frac{2}{\beta} \tag{5}$$

The $L(\beta)$ function is subject to certain limitations. The requirements for the hyperplane to correctly classify all Xi training examples are therefore determined by the constraints. Minimizing this function maximizes the M value.

$$\min_{\beta,\beta_0} L(\beta) = \frac{1}{2}\beta^2 subject\ to\ y_i(\beta^T x_i + \beta_0) \tag{6}$$

After applying the SVM machine learning algorithm using a separating hyperplane on a given dataset, it gets classified into various categories depending on the dimension. Using this, we determine the prediction accuracy. While calculating, a confusion matrix is created which is a technique for reiterating the efficiency of the algorithm and is represented as a true positive, true negative, false positive and false negative value. Based on the values obtained from the matrix, accuracy is calculated using the function accuracy score which is given by the amount of correct predictions. It could also be a fraction. This function gives the accuracy of a subset in a multilabel classification. This accuracy is usually considered as 1.0 for all the set of labels which has correctly been predicted as true and if not, it is computed as 0 (Gereon 2018).

$$accuracy(Y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \tag{7}$$

Here, the accuracy y_i, also known as the true value, is calculated by estimating the fraction of right predictions in n samples and the ith sample is being considered.

Naïve Bayes algorithm

It is on the basis of the Bayes' theorem that the Naïve Bayesian classifier (Mahendru 2019) is formed, while also assuming that there is independence between the predictors. It is very easy to build a Naïve Bayesian model, and it is extremely useful for datasets of a large size since there is no complex parameter estimation of an iterative nature. The performance of the Naïve Bayesian classifier is often surprisingly good despite it being quite a simple model, and it is known to outperform classification methods of a more sophisticated nature quite often, which is why it is so widely used (Gereon 2018;

www.analyticsvidhya.com/blog/2019/06/introduction-powerful-bayes-theorem-data-science). Using Bayes' theorem, we can determine the probability at which an event is likely to occur, assuming that we have another event's probability which has already occurred. The mathematical statement of the Bayes' theorem is as follows:

$$P(A|B) = P(B|A)\frac{P(B)}{P(A)} \tag{8}$$

The Bayes' theorem can be applied in the following way, with regard to our dataset:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \tag{9}$$

Here, y refers to the class variable and X refers to an n sized dependent feature vector wherein:

$$X = (x_1, x_2, \ldots, x_n)$$

In simple terms, given that event B has already occurred, we try to find the probability of the occurrence of event A. Event B may be called the evidence in this case. $P(A)$ is the prior probability of event A, which refers to the event's probability before some evidence is brought to light. The evidence indicates an attribute value of some unknown instance, which in this case is event B. $P(\mathrm{A}|\mathrm{B})$ refers to event A's probability after seeing the evidence.

It is now time to make an assumption of a naïve nature with regard to the Bayes' theorem, i.e. that the features are all independent of one another. The evidence is now split into its independent parts. Supposing we have event A and event B, which are independent of one another, then:

$$P(A, B) = P(A)P(B) \tag{10}$$

We would now obtain the following result:

$$P(y|x_1, \ldots x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)} \tag{11}$$

Another way to express this would be:

$$P(y|x_1, \ldots x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)\ldots P(x_n)} \tag{12}$$

The term of the denominator can be removed as it would remain constant for any given input, which would give us the following:

$$P(y|x_1, \ldots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y) \tag{13}$$

The next step is to create a model for the classifier. In order to do so, we determine the probability for a particular input set having all possible values for y (the class variable), following which the output with the maximum probability is picked up. Mathematically, this could be expressed as follows:

$$y = argmax_y\, P(y)\prod_{i=1}^{n} P(x_i|y) \tag{14}$$

Ultimately, we need to calculate $P(y)$ and $P(x_i|y)$. $P(y)$ represents the class probability, and $P(x_i|y)$ represents the conditional probability.

K-nearest neighbor

KNN (Introduction to K-Nearest Neighbours – A Powerful Machine Learning Algorithm 2018) is a non-parametric algorithm used for classification and regression. The input comprises of closest significant workspace and the output depends on the classification or regression of the KNN algorithm. The classification is carried out based on the majority of the votes. The object is allocated to the class and mostly among its k closest neighbors. When the value of k is equal to one, the object is allocated to the one nearest to it. The property value is found to be the output for KNN regression (Gereon 2018; www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/). The output is the mean of values of its neighbors present around the example values (see Figure 3).

Steps to implement KNN model:

1. Import the dataset.
2. Set the initial value of k.
3. To determine the value of K:

    1. The foremost problem in KNN algorithm is to determine the value of K. A lesser value of K implies that the noise will have a greater impact on the output,

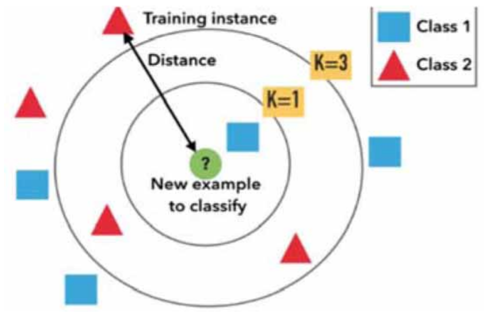


Figure 3 | KNN classification (https://blog.usejournal.com/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7).

i.e. there is a possibility of over fitting. A greater value of K increases the computational cost and the logic behind the KNN algorithm is lost. The easiest method to determine the value for k is $k = \sqrt{n}$.

2. Cross validation is carried out to optimize the results. By this, different values of K can be determined. A model which provides the best accuracy can be considered as an ideal choice.
3. Depending on each instance, it is better to run every value of k and test to obtain optimal results.
4. To obtain the output class, repeat from one to the total number of data points.
   1. The distance between test data and training data in each row needs to be calculated. Euclidean distance is the most popular method used as a distance metric here. The formula for Euclidean distance is as follows:

$$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2} \quad (15)$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \quad (16)$$

   2. Once the distances are calculated using the formula, the values are arranged in ascending order.
   3. Obtain the k rows from the array.
   4. Obtain the recurrent classes of these rows.
   5. Predicted class is returned.

The other distances that can be used in KNN are Manhattan, Chebyshev and Cosine which are used based on a dataset and its application (Importance of Distance Metrics in Machine Learning Modelling n.d).

Manhattan is used where distance is measured as rectilinear distance. Chebyshev distance is between the centers of the squares. Cosine similarity measures the similarity between two vectors of an inner product space. It is often used to measure document similarity in text analysis.

So, based on our application which requires plotting a line between two data points in a plane, Euclidean distance is chosen and is the most popular.

Deep neural network

A perceptron (Perceptron: The Artificial Neuron 2018; What the Hell is Perceptron 2017) is a known artificial neuron forming the neural system. Motivated by Walter Pitts & Warren McCulloch, researcher Frank Rosenblatt developed Perceptron in the 1960s.

So how do perceptrons function? Few inputs are given to the perceptron such as x_1, x_2, and a single binary output is produced.

As illustrated in Figure 4, there are three input sources for the perceptron which are x_1, x_2, and x_3. Algebraically:

$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \le \text{threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{threshold} \end{cases}$$

This is how a perceptron functions!

Neural network is a technology used to specifically imitate the functioning of the human brain and to understand pattern recognition in addition to passing input through different layers of simulated neural connections.

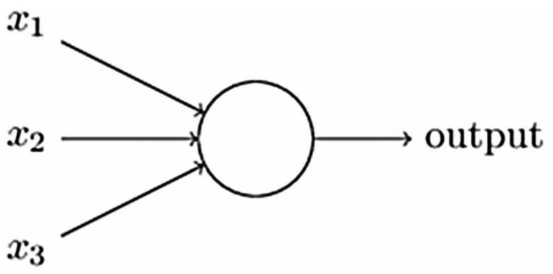


Figure 4 | A perceptron (https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d).

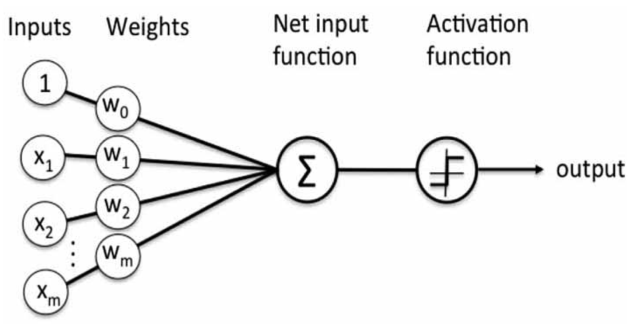


Figure 5 | Perceptron in ANN (https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53).

Artificial neural networks (Gereon 2018) have an input layer, at least one hidden layer in between and an output layer. In a process referred as feature hierarchy, each layer carries out specific sorting and ordering types. A neural network deals with unlabeled or unstructured data which is one of the significant usages. Figure 5 shows the perceptron in ANN.

Suppose we have the network as shown in Figure 6. As previously specified, the input neurons present in the input layer is known as the left-most layer while the right-most layer consists of the outer neurons present in the outer layer. The middle layer is the hidden layer which does not contain neurons of input or neurons of output. The term 'hidden' may sound somewhat secret – the first time we heard the term we thought it had to have some history behind it philosophically or otherwise but in reality, it is just 'not an input or output'. The above system contains a single hidden layer unlike few systems having various masked layers. For example, Figure 7 depicts an arrangement of two hidden layers within the four-layer arrangement.

To some extent surprisingly, and for dependable reasons, such multilayer networks are rarely referred to as multilayer perceptrons (MLPs), irrespective of whether they consist of sigmoid neurons or not (Gereon 2018).

The input and output layer plan in a system is often straightforward. While the idea of a neural system's input and output layers is often direct, the outline of the hidden layers can be considerably crafted. Precisely, with a few straightforward reliable guidelines, it is impractical to complete the outline procedure for the hidden layers. Instead, experts in neural systems have designed several outline techniques for the concealed layers, enabling people to perform out of their need. These heuristics, for example, can be used to determine how to exchange the quantity of hidden layers for the time needed to formulate the system.

Up to now, we have discussed neural arrangements where the one layer output is used as the information to the succeeding layer, which can be called feedforward neural systems. This points towards the system not having any circles, and data is continuously simulated forward, never backed up. On the odds that circles would be present, we would be winding up with conditions where the involvement to the 'σ' work was based on yield. That would be hard to comprehend, so we do not allow that kind of circle.

Neural networks in the gradient descent algorithm can take their weights and biases. How can we handle the cost

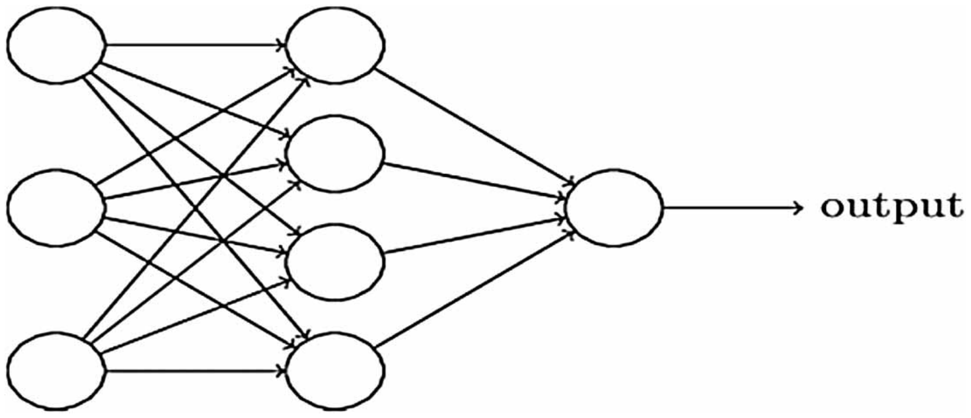


Figure 6 | ANN with only one hidden layer (http://inspirehep.net/record/1507419/plots).

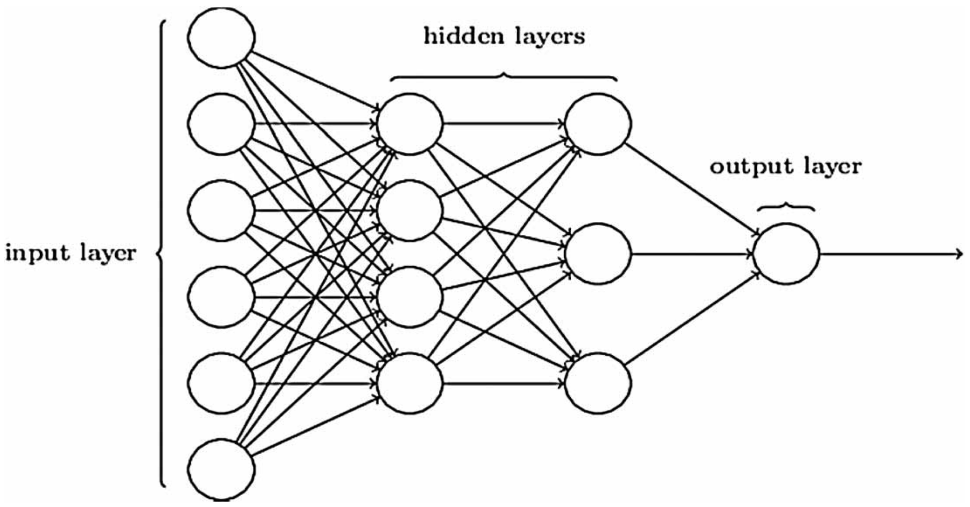


Figure 7 | ANN with two hidden layers (https://www.researchgate.net/figure/Architecture-of-two-hidden-layers-ANN-model-having-five-nodes-in-the-first-and-second_fig3_282939984).

work slope? The significant downside in neural networks is with processing the cost work slope. Error-backpropagation is a quicker way to deal with such slope processing.

In 1986, a paper published by Rumelhart *et al.* (1986) established the significance of error-backpropagation calculation which had been initially anticipated in the 1970s. The paper includes instances of different networks wherein error-backpropagation was proved useful and it outperforms earlier work where several additional algorithms were doing similar calculations.

The most important factor to understand the model of the neural network is cost change given by an articulation for the fractional subsidiary $\partial C/\partial w$ at the cost of work C as for any weight w (or predisposition b) in the system. The articulation illustrates the way biases and weights quickly change. Thus, the articulation is quite confounding, though it has a pleasing factor to it, with each constituent possessing a distinguishing, natural understanding. Therefore, error-backpropagation (Gereon 2018) is not just a learning concept, but gives an in-depth knowledge of how metrics are changed for the system's behavior.

Deep learning (Building Deep Learning Model Using Keras 2018) nowadays has achieved unparalleled success in a variety of tasks of ML or artificial intelligence, such as computer vision, NLP (natural language processing) and reinforcement learning. One main technique in deep learning is deep neural network. A typical DNN model is based on hierarchy of composition of linear functions and a given nonlinear activation function (Daniel & Huerta 2018).

Model

Given n, $m \geq 1$, the first ingredient in defining a deep neural network (DNN) is (vector) linear functions of the form:

$$\Theta{:}Rn \rightarrow Rm \tag{17}$$

as $\Theta(x) = W\,x + b$ where $W = (wij) \in R\ m \times n,\ b \in R\ m.$

The second main ingredient is an activation function which is non-linear, usually denoted as:

$$\sigma{:}R \rightarrow R \tag{18}$$

Applying the function to each component, we can extend this naturally to:

$$\sigma{:}Rn7 \rightarrow Rn \tag{19}$$

Given $d,\ c,\ k \in N+$ and:

$$n1,\ \ldots,\ nk \in N \text{ with } n0 = d,\ nk + 1 = c \tag{20}$$

A general DNN from $R\ d$ to $R\ c$ is given by:

$$f(x) = fk(x),$$
$$f^{\grave{}}(x) = [\Theta^{\grave{}} \circ \sigma](f^{\grave{}} - 1(x))^{\grave{}} = 1{:}k \tag{21}$$

with $f\,0\,(x) = \Theta(x)$. The following more concise notation is often used in computer science literature:

$$f(x) = \Theta k \circ \sigma \circ \Theta k - 1 \circ \sigma \cdots \circ \Theta 1 \circ \sigma \circ \Theta 0\,(x) \tag{22}$$

here $\Theta i : R\ ni \to R\ ni + 1$ are linear functions as defined in Equation (17).

This type of DNN is called a DNN $(k+1)$ layer, and is said to have k hidden layers. Unless otherwise stated, all layers mean hidden layers.

The size of this DNN is $n1 + \cdots + nk$.

In this project, we consider RELU as a special activation function (rectified linear unit), defined by:

$$\begin{aligned} &\text{RELU}{:}R7 \to R, \\ &\text{RELU}(x) = \max(0, x),\ \ x \in R \end{aligned} \tag{23}$$

An RELU DNN with k hidden layers may be written as:

$$f(x) = \Theta k \circ \text{RELU} \circ \Theta k - 1 \circ \text{RELU}\ \ \cdots \circ \Theta 1 \circ \text{RELU} \circ \Theta 0\,(x) \tag{24}$$

The reason for choosing RELU within hidden layers is that at a time only a few neurons are activated which makes the network sparse, efficient and easy for computation. In other words, RELU learns faster than other activation functions which are Sigmoid and Tanh (Activation Functions in Neural Networks n.d).

We propose using deep neural network or deep learning. What distinguishes deep networks from other networks is the use of stacking multiple layers of neurons in appropriate architectures. DL models can take in huge quantities of values and metrics and can often associate successfully to new instances.

Deep learning is best used to describe the features and patterns that can represent the data. The layers are hierarchical with the top layers learning high level features (edges), and the low-level layers learning more data specific features. Deep learning is used in many applications: speech recognition, text classification, and also predicting time series data. Figure 8 shows a deep neural network structure.

System architecture

Having discussed the machine learning algorithms mathematical model used in implementation, the architecture of an early flood warning system using a deep neural network prediction system is shown in Figure 9. DFD and use case diagrams of the system are developed as shown in Figures 10 and 11.

In our work, the major step is to find a suitable dataset which is then split into training and testing sets. Using this, the model can then be defined and trained, following which the results can be analyzed. If a high bias is seen, then the model is not suitable and hence must be redefined. If so, then all the corresponding steps must be redone accordingly until a lower bias is seen for a given model. Once this is achieved, the model is ready to be tested with real life data, and the metrics for it will be inspected accordingly. Finally, once all of these steps are performed to satisfaction, the model is ready to be deployed.

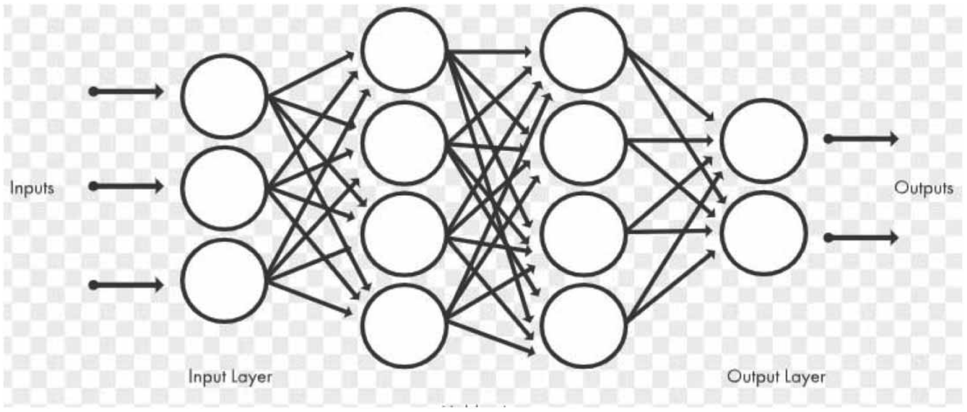


Figure 8 | DNN structure (https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37).

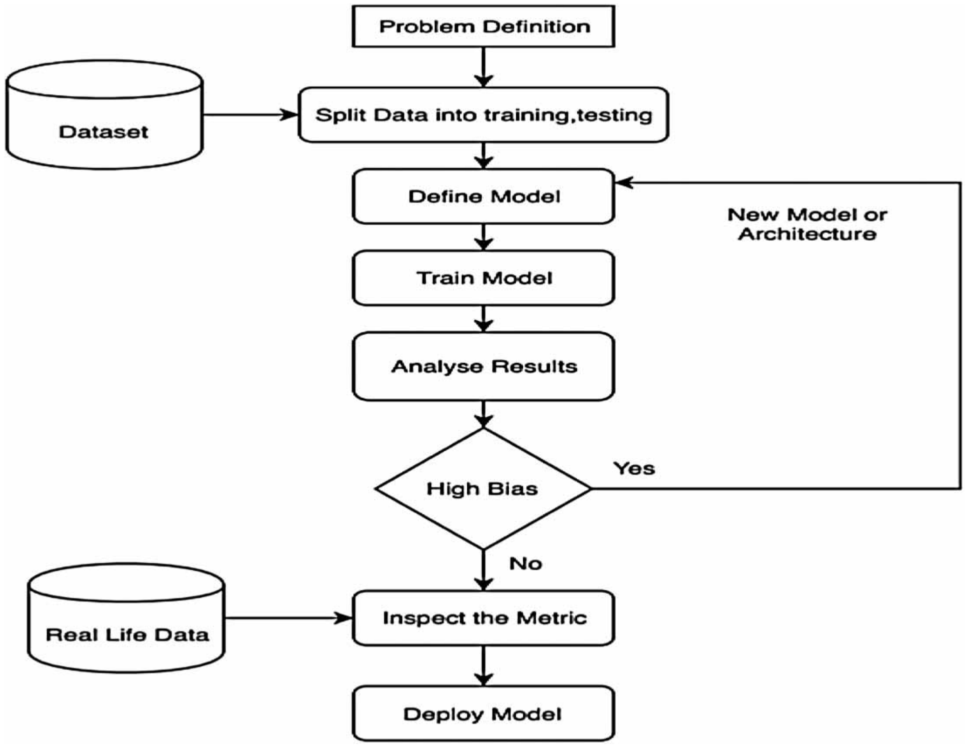


Figure 9 | System architecture.

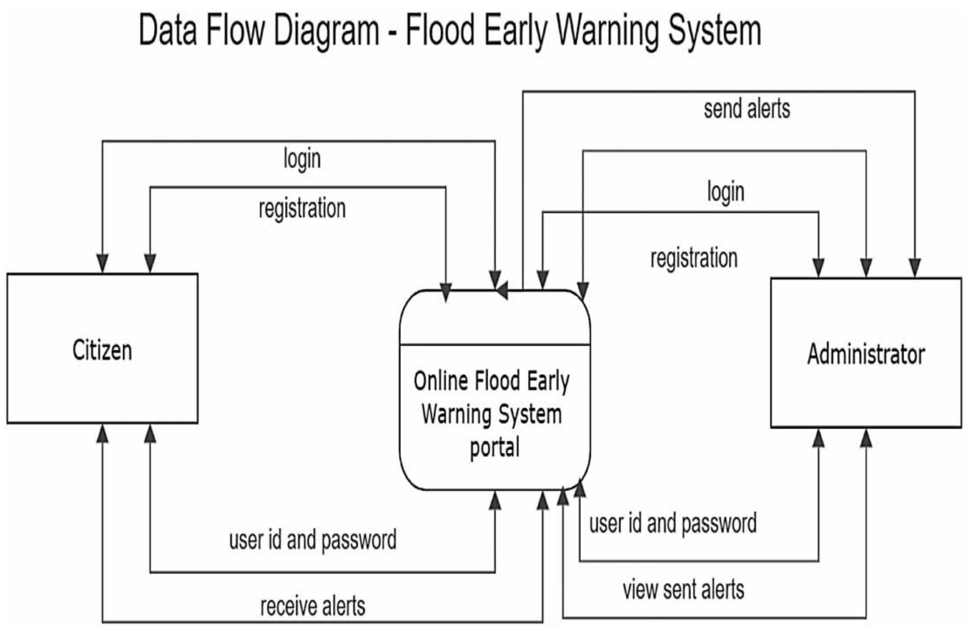


Figure 10 | Data flow diagram.

In our work, the data flow diagram shows the flow of input and output data between:

- citizen and online flood early warning system portal;
- administrator and online flood early warning system portal.

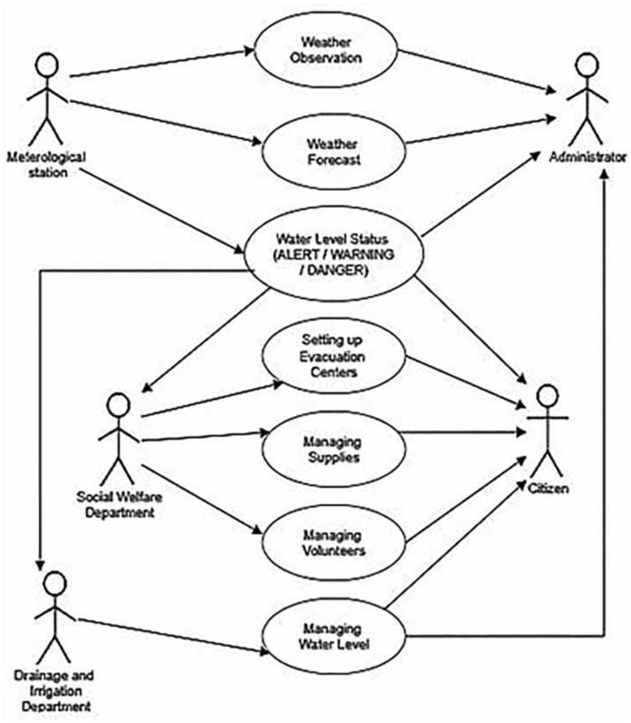

Weather Observation
Weather Forecast
Meterological station
Administrator
Water Level Status (ALERT / WARNING / DANGER)
Setting up Evacuation Centers
Managing Supplies
Managing Volunteers
Managing Water Level
Social Welfare Department
Citizen
Drainage and Irrigation Department

Figure 11 | Use case diagram.

Figure 11 depicts a detailed use case diagram involving five actors: meteorological station, social welfare department, drainage and irrigation department, citizen and administrator. It shows that the possible actions the meteorological station can perform are weather observation, weather forecasting and updating the water level status (ALERT/WARNING, DANGER), all of which are received and cataloged by the administrator. On receiving the water level status update, the actions performed by the social welfare department are setting up evacuation sites, managing supplies and managing volunteers. Also, the actions performed by the drainage and irrigation department on receiving the water level status update is to manage the water level. Regular updates on all of these actions are sent to citizens in order to give them ample time to prepare for the eventuality of a flood and know how best to handle it.

RESULTS AND DISCUSSION

The main focus of this work is on developing an early flood warning system using a deep neural network. Python's advanced built in data structures, linked with dynamic composing and dynamic binding to associate existing components together, are used. Some of the libraries in Python used for our work are Keras, Pandas and Numpy. A database is created containing the flood rainfall data. The reason for choosing python for implementing machine learning is that Python has many libraries for every need of your AI project. A few names include 'Numpy' for scientific computation, 'Scipy' for advanced computing and 'Keras' for machine learning. Python is a completely open source with a great community. There is a host of resources available which can get any developer up to speed in no time.

Selection of dataset

For our research, India was ideal for our project. Following this, all the states and union territories in the country were researched based on the frequency of occurrence of major floods, along with the corresponding death toll as well as damage to property and infrastructure. On doing so, it was found that Bihar is one of the most flood-prone states in the country, with about 73% of its total surface area getting flooded annually, as quoted by the International Water Management Institute (IWMI). Orissa also had a very concerning frequency of flood occurrence across its various districts and hence was included in the regions to be further studied. Once the states were decided, their individual districts were then studied further for the creation of a decomposed dataset for Bihar and Orissa. Since both Bihar and Orissa are fairly large states, with their number of districts at 38 and 30 respectively, it was decided that consolidating further was necessary (India Water Portal n.d). Hence it was finally decided that the ten most flood affected districts of each state would be considered for this project. The districts selected from the state of Bihar were: Khagaria, Madhubani, Patna, Samastipur, Sitamarhi, Bhagalpur, Darbhanga, Purba Champaran, Pashchim Champaran, and Katihar. The districts selected from the state of Orissa were: Baleshwar, Bhadrak, Cuttack, Jagatsinghapur, Jajapur, Kalahandi, Kendrapara, Koraput, Puri, and Sambalpur.

Once the regions were decided, the next step was to decide what features needed to be considered for the prediction of flood occurrence. Aside from the state and district, the various columns present in this dataset are year, month, maximum temperature, minimum temperature, rainfall and flood occurrence. The reason why month is also considered here is because in India, rainfall tends to be much higher during the monsoon season, usually between the months of June and September. Finally, the data was taken over a period of 13 years, from 1990 to 2002. Figure 12 shows the screenshot of a sample dataset used for our work.

Metrics for algorithmic comparison

The implementation of the different machine learning algorithms such as SVM, Bayes, KNN and deep neural network are compared to identify the best algorithm for the prediction of the occurrence of flood. Before the results, a brief explanation about the various metrics used for comparative analysis is given below.

Confusion matrix

This is a binary classifier. A confusion matrix can be of any size depending upon the different number of parameters inputted (labels in our case). The confusion matrix in our case is a 2×2 matrix.

TP	FN
FP	TN

where TP = true positive; FN = false negative; FP = false positive; TN = true negative.

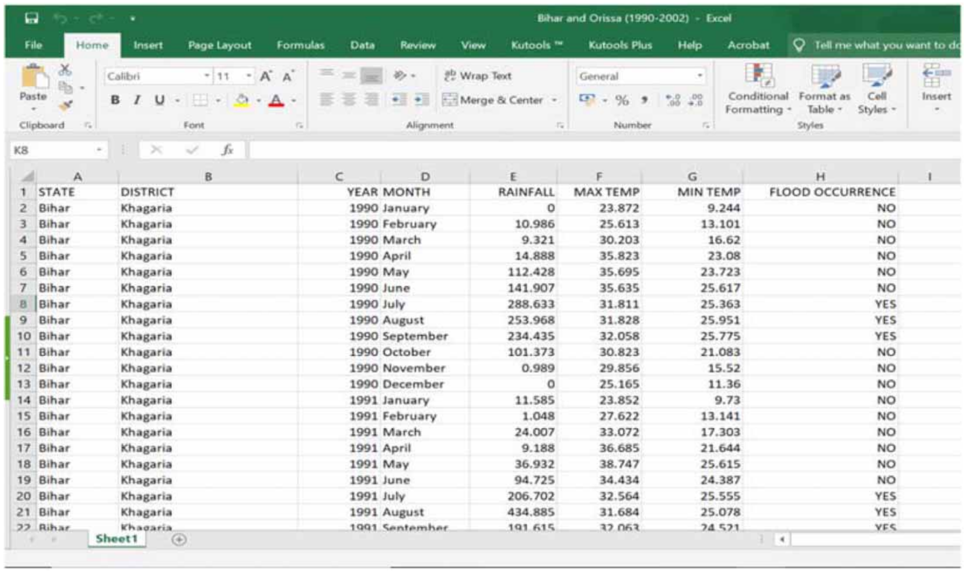
TP and TN denote the number of instances which have been correctly classified as no flood occurrence and flood occurrence respectively. FP and FN signify the number of instances which have been wrongly classified as no flood occurrence and flood occurrence respectively.

Precision and recall

The success of prediction is computed by means of precision-recall where classes are imbalanced. The relevancy of the result is expressed as precision whereas the number of true relevant results returned is expressed as recall. A low false positive rate and low false negative rate result is related to high precision and high recall respectively.

Precision: Termed as the positive predictive value is calculated as given below

$$Precision = \frac{TP}{TP + FP}$$



	A	B	C	D	E	F	G	H
1	STATE	DISTRICT	YEAR	MONTH	RAINFALL	MAX TEMP	MIN TEMP	FLOOD OCCURRENCE
2	Bihar	Khagaria	1990	January	0	23.872	9.244	NO
3	Bihar	Khagaria	1990	February	10.986	25.613	13.101	NO
4	Bihar	Khagaria	1990	March	9.321	30.203	16.62	NO
5	Bihar	Khagaria	1990	April	14.888	35.823	23.08	NO
6	Bihar	Khagaria	1990	May	112.428	35.695	23.723	NO
7	Bihar	Khagaria	1990	June	141.907	35.635	25.617	NO
8	Bihar	Khagaria	1990	July	288.633	31.811	25.363	YES
9	Bihar	Khagaria	1990	August	253.968	31.828	25.951	YES
10	Bihar	Khagaria	1990	September	234.435	32.058	25.775	YES
11	Bihar	Khagaria	1990	October	101.373	30.823	21.083	NO
12	Bihar	Khagaria	1990	November	0.989	29.856	15.52	NO
13	Bihar	Khagaria	1990	December	0	25.165	11.36	NO
14	Bihar	Khagaria	1991	January	11.585	23.852	9.73	NO
15	Bihar	Khagaria	1991	February	1.048	27.622	13.141	NO
16	Bihar	Khagaria	1991	March	24.007	33.072	17.303	NO
17	Bihar	Khagaria	1991	April	9.188	36.685	21.644	NO
18	Bihar	Khagaria	1991	May	36.932	38.747	25.615	NO
19	Bihar	Khagaria	1991	June	94.725	34.434	24.387	NO
20	Bihar	Khagaria	1991	July	206.702	32.564	25.555	YES
21	Bihar	Khagaria	1991	August	434.885	31.684	25.078	YES
22	Bihar	Khagaria	1991	September	191.615	32.063	24.521	YES

Figure 12 | Flood occurrence dataset (India Water Portal n.d).

Recall: Also called as sensitivity is calculated as given below.

$$Recall = \frac{TP}{TP + FN}$$

Accuracy

Based on the confusion matrix created, accuracy computed as an accuracy score function which is either by default the fraction or the count (normalize = false) of correct predictions. Accuracy is calculated as follows:

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

F1 score

This is a measure of the test's accuracy where it considers both precision and recall. This is the harmonic average of precision and recall where F1 reaches its best value at 1 and worst at 0. The F1 score conveys the balance between precision and recall:

$$F1\ score = 2 * \frac{\mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

MCC

Matthews correlation coefficient considers all four divisions of the confusion matrix when calculated. MCC lies within the range −1 to +1 where a model with a positive score is considered to be perfect whereas the negative score is poor. This makes this metric really useful as it is easy to interpret (Gereon 2018):

$$MCC = \frac{TP * \mathrm{TN} - \mathrm{FP} * \mathrm{FN}}{\sqrt{(TP + FP) * (\mathrm{TP} + \mathrm{FN}) * (\mathrm{TN} + \mathrm{FP}) * (\mathrm{TN} + \mathrm{FN})}}$$

Performance of machine algorithms

For our work, we have considered a dataset from Indian Water Portal (n.d.) comprising rainfall, flood occurrence and temperature of Bihar and Orissa states. Further to that, based on research, only 10 districts were selected for the states of Bihar and Orissa for flood occurrence. The dataset was then decomposed for the states of Bihar and Orissa comprising ten districts each of which constituted 3,000 datasets having parameters such as rainfall, minimum temperature, maximum temperature and flood occurrence. With regard to the deep neural network, we have considered three inputs followed by three hidden layers and an output layer. In here, each hidden layer possesses 10 neurons or nodes. These input parameters were trained over 10,000 epochs, thereby producing the appropriate classification as flood occurrence (1) or no flood occurrence (0). During the training, weights were adjusted based on the error produced by applying error backpropagation towards achieving the predicted output. The activation functions used are LeakyRelu and Softmax within a hidden layer and an output layer respectively.

In deep learning, RELU is predominantly used over Tanh and other activation functions within hidden layers. RELU is used for activating the neurons at a time only making the network sparse, efficient and easy for computation. In simple words, RELU learns much faster than the sigmoid and Tanh functions.

In terms of output, SoftMax is predominantly used at the output neuron for handling classification problems (Activation Functions in Neural Networks n.d).

Towards prediction on the preprocessed dataset, different machine learning algorithms such as SVM, Bayes, KNN and deep neural networks are employed towards computing the prediction accuracy. The results of prediction accuracy and other related metrics for each algorithm are given in Tables 1–4.

From the analysis of different machine learning algorithms, it is clear that for SVM, Bayes, KNN and DNN,

Table 1 | SVM performance metrics

Detailed accuracy terms	Value
Accuracy	85.57%
Precision	0.88
Recall	0.94
F1 Score	0.9
MCC	0.4

Table 2 | Naïve Bayes performance metrics

Detailed accuracy terms	Value
Accuracy	87.01%
Precision	0.96
Recall	0.87
F1 Score	0.91
MCC	0.64

Table 3 | KNN performance metrics

Detailed accuracy terms	Value
Accuracy	85.73%
Precision	0.91
Recall	0.90
F1 Score	0.91
MCC	0.50

Table 4 | DNN performance metrics

Detailed accuracy terms	Value
Accuracy	91.18%
Precision	0.95
Recall	0.93
F1 Score	0.95
MCC	0.64

precision and recall is high which indicates that predicted results are all accurate and positive. There are very few inaccurate and negative results which are false positive and false negative.

Mathew's correlation coefficient should lie between -1 and $+1$ which indicates the quality of binary classification. A positive value indicates perfect classification. In regards to the F1 score, the test's accuracy is computed, which again is near 1, indicating near perfect.

Finally, the accuracy of prediction is high for DNN (91.18%) as compared to other machine learning algorithms. This indicates that the system is able to provide the best prediction with a deep neural network as compared with other machine learning algorithms with very few inaccurate predictions, see Tables 1–4.

Figure 13 illustrates the mean values of the accuracies of the DNN, Naïve Bayes, SVM and KNN algorithms taken over a range of 20 readings each. It can be clearly seen that DNN outperforms the other algorithms with an accuracy of 89.71%. The Naïve Bayes comes second with an accuracy of 87.86%, followed by SVM and KNN with accuracies of 87.05 and 86.25% respectively. On average it is shown that DNN gives the best prediction with very few inaccurate results compared to other machine learning algorithms.

Figure 14 illustrates the standard deviation values of the accuracies of the DNN, Naïve Bayes, SVM and KNN algorithms taken over a range of 20 readings each. As can be seen, DNN shows the most amount of dispersion within the set of data values.

CONCLUSION AND FUTURE WORK

India contributes one-fifth of the world's global deaths due to floods and is the world's worst affected country after Bangladesh. The Indian rainfall season is mainly from June to September and accounts for nearly 75% of the total Indian rainfall per year. Much work has been carried out by employing machine learning algorithms such as ANN for flood prediction. Most of the systems employed ANN with a single hidden layer for prediction of flood with parameters such as rainfall, temperature, water flow, water level and humidity. One system was developed using a deep neural network where stream flow was taken into consideration for the prediction of flood.

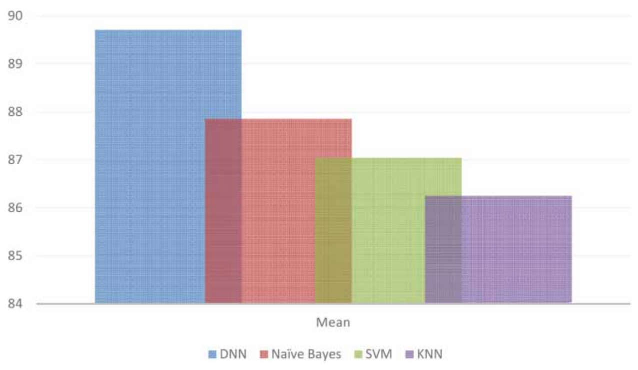


Figure 13 | Comparative analysis of algorithms (mean).

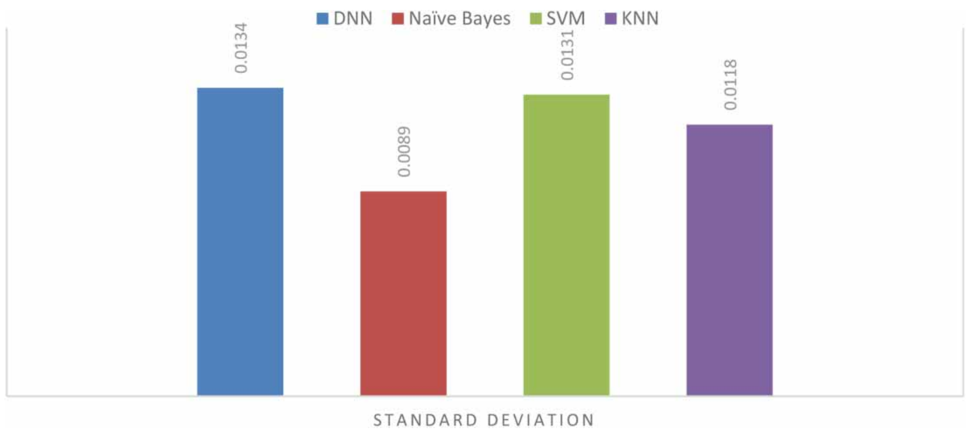


Figure 14 | Comparative analysis of algorithms (standard deviation).

The challenge in all these system is that most used traditional ANN and with the advent of the deep neural network, we can predict the possibility of flood occurrence with higher accuracy based on rainfall and temperature.

This research work employed a deep neural network for forecasting the prediction of occurrence of flood based on temperature and rainfall. Based on the prediction of probability of flood occurrence, an alert can be provided for evacuation which can save human lives and property. The dataset consisted of large amounts of observed rainfall data and collecting factors such as minimum and maximum temperature and flood occurrence. On comparing the accuracy obtained from four different algorithms, the results indicate that the deep neural network can be efficiently used for flood forecasting. The work carried out here clearly shows that the DNN provides better accuracy when compared with other algorithms such as SVM, KNN and Naïve Bayes with a precision of 89.71%.

The limitation of the research work is that we have a dataset from 1990 to 2002 only for validating the machine learning model towards flood prediction. The most recent dataset is not available which could result in better accuracy than achieved, which is not a major issue.

Many aspects from the future work can be tackled from this research. First, data distribution imbalance was not considered. With the advancement of neural networks for prediction, further work can be implemented by considering a most recent dataset which could achieve higher accuracy. Second, other topographical factors including flood water level can be considered in order to predict floods more precisely for different cities/states in India. Last, by including standard machine learning paradigms, the results obtained open the field for further research and development of new methods.

REFERENCES

Abdullahi, S. I., Habaebi, M. H. & Malik, N. A. 2018 Flood Disaster Warning System on the go. In: *2018 7th International Conference on Computer and Communication Engineering (ICCCE)*. IEEE, Romania, pp. 258–263.

Activation Functions in Neural Networks n.d. Available from: www.geeksforgeeks.org/activation-functions-neural-networks/.

Adams, T. E. & Pagano, T. C. (eds) 2016 *Flood Forecasting: A Global Perspective*. Academic Press, UK.

Bande, S. & Shete, V. 2017 Smart flood disaster prediction using IoT and neural networks. In: *Proceedings of IEEE International Conference on Smart Technologies for Smart Nation*, Bangalore, India.

Building Deep Learning Model using Keras 2018 Retrieved from: https://towardsdatascience.com/building-a-deep-learning-model-using-keras-1548ca149d37.

Camps-Valls, G. 2009 Machine learning in remote sensing data processing. In: *Proceedings of IEEE International Conference on Machine Learning for Signal Processing*, Grenoble, France.

Chuntian, C. & Chau, K. W. 2002 Three-person multi-objective conflict decision in reservoir flood control. *Eur. J. Oper. Res.* **142** (3), 625–631.

Color Classification of images with Support Vector Machine 2018 Retrieved from: https://medium.com/deepvision/color-classification-with-support-vector-machine-755360c96037.

Daniel, G. & Huerta, E. A. 2018 Deep neural networks to enable real-time multi messenger astrophysics. *Phys. Rev. D* **97** (4), 1–17.

Duncan, A., Keedwell, E., Djordjevic, S. & Savic, D. 2013 *Machine Learning-Based Early Warning System for Urban Flood Management*. Retrieved from: https://pdfs.semanticscholar.org/9b6b/ee56f3c435b674171f3a954794ce6a7bd72c.pdf?_ga=2.60033581.1665221806.1561186936-775818349.1554308375.

FLEWS n.d. Retrieved from: https://darpg.gov.in/sites/default/files/70.%20 Flood%20Early%20Warning%20SystemFLEWS-Documentation-Final.pdf.

Fotovatikhah, F., Herrera, M., Shamshirband, S., Chau, K. W., Faizollahzadeh Ardabili, S. & Piran, M. J. 2018 Survey of computational intelligence as basis to big flood management: challenges, research directions and future work. *Eng. Appl. Comput. Fluid Mech.* **12** (1), 411–437.

Ganguly, K. K., Nahar, N. & Hossain, B. M. 2019 A machine learning-based prediction and analysis of flood affected households: a case study of floods in Bangladesh. *Int. J. Disaster Risk Reduct.* **34**, 283–294.

Gereon, A. 2018 *Hands-on Machine Learning with Scikit-Learn and Tensor Flow*. O' Reily Media Inc., USA.

Ghorbani, M. A., Kazempour, R., Chau, K. W., Shamshirband, S. & Taherei Ghazvinei, P. 2018 Forecasting pan evaporation with an integrated artificial neural network quantum-behaved particle swarm optimization model: a case study in Talesh, northern Iran. *Eng. Appl. Comput. Fluid Mech.* **12** (1), 724–737.

Han, S. & Coulibaly, P. 2017 Bayesian flood forecasting methods: a review. *J. Hydrol.* **551**, 340–351.

Hossain, M. E., Turna, T. N., Soheli, S. J. & Kaiser, M. S. 2014 Neuro-fuzzy (NF)-based adaptive flood warning system for Bangladesh. In: *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*. IEEE, Japan, pp. 1–5.

Importance of Distance Metrics in Machine Learning Modelling n.d. Retrieved from: https://towardsdatascience.com/importance-of-distance-metrics-in-machine-learning-modelling-e51395ffe60d.

India Water Portal n.d Retrieved from: https://www.indiawaterportal.org/data.

Introduction to K-Nearest Neighbours – A Powerful Machine Learning Algorithm 2018 Retrieved from: www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering.

Khalaf, M., Hussain, A. J., Al-Jumeily, D., Fergus, P. & Idowu, I. O. 2015 Advance flood detection and notification system based on sensor technology and machine learning algorithm. In: *International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 105–108.

Liu, F., Xu, F. & Yang, S. 2017 A flood forecasting model based on deep learning algorithm via integrating stacked auto-encoders with BP neural network. In: *2017 IEEE Third International Conference on Multimedia Big Data (BigMM)*. IEEE, Taiwan, pp. 58–61.

Mahendru, K. 2019 *An Introduction to the Powerful Bayes' Theorem for Data Science Professionals*. Retrieved from: www.analyticsvidhya.com/blog/2019/06/introduction-powerful-bayes-theorem-data-science.

MAP Rain n.d. *Helping to Monitor and Predict Flooding Events*. Retrieved from: www.meniscus.co.uk/meniscus-analytics-case-studies/flooding/.

Mitra, P., Ray, R., Chatterjee, R., Basu, R., Saha, P., Raha, S. & Saha, S. 2016 Flood forecasting using internet of things and artificial neural networks. In: *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference*. IEEE, Canada, pp. 1–5.

Moazenzadeh, R., Mohammadi, B., Shamshirband, S. & Chau, K. W. 2018 Coupling a firefly algorithm with support vector regression to predict evaporation in northern Iran. *Eng. Appl. Comput. Fluid Mech.* **12** (1), 584–597.

Opella, J. M. A. & Hernandez, A. A. 2019 Developing a flood risk assessment using support vector machine and convolutional neural network: a conceptual framework. In: *2019 IEEE 15th International Colloquium on Signal Processing & Its Applications*, pp. 260–265.

Perceptron: The Artificial Neuron 2018 Retrieved from: https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d.

Rumelhart, D., Hinton, G. & Williams, R. 1986 Learning representations by back-propagating errors. *Nature* **323**, 533–536.

Sankaranarayanan, S. & Mookherji, S. 2019 SVM-based traffic data classification for secured IoT-based road signalling system. *Int. J. Intell. Inf. Technol. (IJIIT)* **15** (1), 22–50.

Wang, W. C., Chau, K. W., Qiu, L. & Chen, Y. B. 2015 Improving forecasting accuracy of medium and long-term runoff using artificial neural network based on EEMD decomposition. *Environ. Res.* **139**, 46–54.

What the Hell is Perceptron 2017 Retrieved from: https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53.

First received 24 June 2019; accepted in revised form 21 October 2019. Available online 14 November 2019