**ABSTRACT:**

In this paper, we present a model for rainfall rate prediction 30 seconds ahead of time using an artificial neural network. The resultant predicted rainfall rate can then be used in determining an appropriate fade counter-measure, for instance, digital modulation scheme ahead of time, to keep the bit error rate (BER) on the link within acceptable levels to allow constant flow of data on the link during a rain event. The approach used in this paper is pattern recognition technique that considers historical rainfall rate patterns over Durban (29.8587°S, 31.0218°E). The resultant prediction model is found to predict an immediate future rain rate when given three adjacent historical rain rates. For our model validation, error analysis via root mean square (RMSE) technique on our prediction model results show that resultant errors lie within acceptable values at different rain events within different rainfall regimes.

**INTRODUCTION**

As the mobile communication industry takes a leap into the world of 5G, the millimeter wave is the way to go towards achievement of large bandwidths and subsequently high data rates. As is forecasted, the internet traffic will reach 1.6 Zettabytes (ZB) by 2018 [1] and one alternative to satisfying this demand for larger bandwidths is by utilizing communication channels using electromagnetic waves from Ka-band and above in conjunction with multi-bit digital modulation schemes in the M-ary QAM and others. Unfortunately, link fading and signal outages due to rain attenuation cannot be disregarded when operating radio links in these higher frequency bands (above 10 GHz) [2], [3]. In the past, several mitigation techniques have been employed to ensure that the channel is available to the user under various channel conditions. These mitigation techniques include space diversity, frequency diversity, power control, adaptive coding modulation (ACM) [4] – [7]. Most of these techniques monitor the signal level on the link and then use an alternative feedback channel back to the transmitter to indicate the state of the channel for an action to be effected as is the case with frequency diversity and power control. However, the feedback signal might itself become a victim of fading, hence lacking integrity itself. Hence, we need to predict the future state of the link, in preparation for the next appropriate measure to counter fading. High values of M in the M-ary QAM system, for instance, guarantee large bandwidth for high speed data and video communication. Regrettably, higher data rates attract higher values of bit error rates causing frequent data resend requests that slow down the transmission

**PROBLEM STATEMENT** A model for rainfall rate prediction using an artificial neural network. It is used to determining an appropriate fade counter-measure, for instance, digital modulation scheme.The prediction model is found to predict an immediate future rain rate when given three adjacent historical (instant) rain rates. In this regard, our model validate error analysis via root mean square technique.

**EXISTING SYSTEM** As the mobile communication industry takes a leap into the world of 5G, the millimeter wave is the way to go towards achievement of large bandwidths and subsequently high data rates. As is forecasted, the internet traffic will reach 1.6 Zettabytes (ZB) by 2018 and one alternative to satisfying this demand for larger bandwidths is by utilizing communication channels using electromagnetic waves from Ka-band and above in conjunction with multi-bit digital modulation schemes in the M-ary QAM and others. Unfortunately, link fading and signal outages due to rain attenuation cannot be disregarded when operating radio links in these higher frequency bands (above 10 GHz) . In the past, several mitigation techniques have been employed to ensure that the channel is available to the user under various channel conditions. These mitigation techniques include space diversity, frequency diversity, power control, adaptive coding modulation (ACM) . Most of these techniques monitor the signal level on the link and then use an alternative feedback channel back to the transmitter to indicate the state of the channel for an action to be effected as is the case with frequency diversity and power control.

**DISADVANTAGES:**

• Weather Prediction is not accurate so result is not perfect in the prediction system.

• Satellite image training procedure is complicated.

• Pre-Processing analysis is not satisfied the Prediction Process

**LITERATURE REVIEW** Many studies on rainfall rate prediction using an artificial neural networks have majorly been based on weekly, monthly and yearly rainfall predictions with more emphasis geared towards meteorology and water management [8]. French et al. [9] used an artificial neural network to forecast two-dimensional rainfall, one hour ahead. Though not very successful, their work increased an understanding in the use of artificial neural networks (ANNs) in investigating complex geophysical processes [9], [10]. In 1995, Michaelides et al. [11] used a neural network in the estimation of missing rainfall data over Cyprus. Prediction was done via daily rainfall observations in the neighboring sites.

In another study, Christodoulou et al. [12] applied an artificial neural network to predict rainfall rate from weather radar data. They used two machine learning classifiers, a selforganizing map (SOM) and the statistical K-Nearest Neighbor (KNN) classifier with radar data as input to the neural network and rain-gauge measurements as outputs during training. The average error rate for predicted rainfall rate was 23%. From their studies, Nayak et al. [10], confirmed that the backpropagation network, in addition to other kinds of neural networks, can be relied upon in rainfall prediction with better results compared to statistical and numerical methods.

**IN “T. S. RAPPAPORT, “MILLIMETER WAVE WIRELESS COMMUNICATIONS: THE RENAISSANCE OF COMPUTING AND COMMUNICATIONS,” INTERNATINOAL CONFERENCE ON COMMUNICATIONS, AUSTRALIA, JUNE, 2014”** Millimeter wave (mmWave) communication has raised increasing attentions from both academia and industry due to its exceptional advantages. Compared with existing wireless communication techniques, such as WiFi and 4G, mmWave communications adopt much higher carrier frequencies and thus come with advantages including huge bandwidth, narrow beam, high transmission quality, and strong detection ability. These advantages can well address difficult situations caused by recent popular applications using wireless technologies. For example, mmWave communications can significantly alleviate the skyrocketing traffic demand of wireless communication from video streaming. Meanwhile, mmWave communications have several natural disadvantages, e.g., severe signal attenuation, easily blocked by obstacles, and small coverage, due to its short wavelengths. Hence, the major challenge is how to overcome its shortcomings while fully utilizing its advantages. In this paper, we present a taxonomy based on the layered model and give an extensive review on mmWave communications. Specially, we divide existing efforts into four categories that investigate physical layer, MAC layer, network layer, and cross layer optimization, respectively. First, we present an overview of some technical details in physical layer. Second, we summarize available literature in MAC layer that pertains to protocols and scheduling schemes. Third, we make an in-depth survey of related research work in network layer, providing brain storming and methodology for enhancing the capacity and coverage of mmWave networks. Fourth, we analyze available research work related to cross layer allocation/optimization for mmWave communications. Fifth, we make a review of mmWave applications to illustrate how mmWave technology can be employed to satisfy other services. At the end of each section described above, we point out the inadequacy of existing work and identify the future work. Sixth, we present some available resources for mmWave communications, including related books about mmWave, commonly used mmWave frequencies, existing protocols based on mmWave, and experimental platforms. Finally, we have a simple summary and point out several promising future research directions Nowadays, more and more bandwidth intensive applications are emerging in daily routines of mobile users (e.g., HDTV, UHDV) [1]. Wireless data traffic is projected to skyrocket 10000 fold within the next 20 years [2]. To tackle this incredible increase, one of the most efficient resolutions is to move the data transmissions into an unused nontraditional spectrum where enormous bandwidths are available, such as millimeter wave (mmWave). The mmWave bands roughly corresponding to frequencies from 30GHz to 300GHz have drawn considerable attention because of huge bandwidth. mmWave communications have several merits compared with existing wireless technologies, which are described as follows. • Extremely wide bandwidths: Compared with existing wireless networks, mmWave communications employ much higher frequencies (30-300GHz) as carrier frequencies. Hence, it has much more abundant spectrum resource (270GHz), making itself quite alluring under the conditions of intensive spectrum. • Small element sizes: Owing to short wavelengths, mmWave devices enable large antenna arrays to be packed in small physical dimension. • Narrow beams: With the same antenna size, it is possible to pack more antenna elements at mmWave frequencies than at microwave. Therefore, the formed beam can be narower, which can further facilitate the development of other applications, such as detection radars. However, mmWave communications also suffer from several drawbacks. Due to much higher carrier frequencies compared to conventional wireless techniques, severe attenuation will occur caused by oxygen absorption, which is shown in Fig. 1. From this figure, it can be observed that in some special bands such as 35GHz, 94GHz, 140GHz, and 220GHz, mmWave propagation experiences relatively small attenuation. Thus, long distance communication can be realized in these mmWave bands, which is well suitable for peer to peer communication. However, in the 60GHz, 120GHz, 180GHz bands, mmWave signals attenuate severely as high as 15dB/km, which are known as “attenuation peak”. In general, these bands are employed by covert network and system for multipath diversity so as to satisfy the requirements of network safety factor. Meanwhile, mmWave signals will experience poor diffraction when encountering blockages owing to the short wavelengths [3]. These two defects significantly shorten the transmission range of mmWave signals and easily bring mmWave links to the disconnected state. Thanks to the rapid progress in complementary metal-oxide-semiconductor (CMOS) radio frequency (RF) integrated circuits [4], [5], beamforming based on large-scale mmWave antenna arrays has been widely exploited to extend the coverage of mmWave networks [6], [7]. In the meantime, interference in mmWave based networks can be substantially cut down based on highly directional beams, rendering mmWave networks noise-limited rather than interference-limited in many cases [8]. Yet, many unresolved problems in the physical layer, medium access control (MAC) layer, and network layer induced by its unique characteristics hamper the realization of mmWave communications to full advantage. In the physical layer, extensive research related to the construction of mmWave channels has been carried out [11]–[13]. For instance, owing to widely divergent propagation features compared to LTE, mmWave channels of 28GHz, 38GHz, 60GHz, and 73GHz covering lineof-sight(LOS) and non-line-of-sight(NLOS) are built for both indoor and outdoor environments, such as building offices and dense urban environments. For mmWave circuit components and antenna design, the non-linearity and phase noise become much more serious due to much higher operating frequencies [14]–[16]. Effective strategies are needed to combat these defects [17]–[20]. With much smaller wavelengthes, large-scale antenna arrays with more than 100 elements are designed, based on which beamforming can be achieved to provide high link gain. Meanwhile, protocols in MAC layer should be redesigned since propagation characteristics in physical layer have changed [21]. Therefore, new designed MAC protocols should be adopted to support highly directional transmission links, extreme low latencies, and high peak data rates [22]–[24]. In terms of network layer, majority of existing research work concentrates on multihop routing and relay placement so as to enhance the coverage and capacity of mmWave networks [25]–[27]. For the actual use cases of mmWave communications, many application scenarios have been enumerated, such as data traffic delivery in dense wearable networks, or object detecting and tracking at centimeter accuracy [28]– [30]. Meanwhile, several survey papers containing different aspects of mmWave technology have been published in recent years [9], [31]–[35]. In [32], the authors discuss the technical challenges such as large-scale attenuation, atmospheric absorption, phase noise, limited gain amplifiers in the design of mmWave frameworks. They have also investigated the critical metrics which can characterize multimedia QoS based on mmWave communications. Finally, they propose a QoS-aware multimedia scheduling scheme to realize the trade-off between performance and complexity. The authors of [33] investigate the deployment strategies of small mmWave cells in urban environments. Research work in [9], [31] explores the utilization of mmWave communications in 5G networks. They have pointed out the feasibility, advantages, and challenges if employing mmWave communications in future wireless networks. Recently, Rappaport et al. [34] present a comprehensive survey of mmWave radio propagation models to date. They carry out a detailed elaboration of various models in terms of path loss model, line-of-sight probability, and building penetration. The authors of [35] make a review of mobile networks based on mmWave communications, including recent channel measurements and models, MIMO, and access and backhaul schemes. In addition, they have also introduced the standardization and deployment efforts for mmWave mobile networks. In comparison to existing surveys in the mmWave field, the outline of the contributions of this paper is presented as: • We conduct a more in-depth and comprehensive analysis and summary of mmWave communications, including physical layer, MAC layer, network layer, cross-layer optimization, and use cases, to enable interested individuals to have a quick and overall insight. • Due to the rapid development of mmWave technology, a large amount of research work on mmWave communications has been completed [36]–[38] these years. Therefore, we incorporate these research efforts into this paper, in order to facilitate the understanding of mmWave development trends. • We provide several use cases (e.g., wearable devices) to illustrate how mmWave communications are employed to satisfy the requirements of other services based on its unique features. • We also present available mmWave resources, which include books about mmWave, commonly used mmWave frequencies, mmWave based protocols, and experimental platforms

**IN “G. O. AJAYI AND E.B.C. OFOCHE, “SOME TROPICAL RAIN RATE CHARACTERISTICS AT ILE-IFE FOR MICROWAVE AND MILLIMETER WAVE APPLICATIONS,” JOURNAL OF CLIMATE AND APPLIED METEOROLOGY, VOL. 23, PP. 562-567, 1984”** Some characteristics of rainfall rate relevant to the evaluation of microwave attenuation due to tropical rainfall are presented, making use of rain rate data collected with a rapid response raingage at Ile-Ife between September 1979 and December 1981. The effect of integration time on the cumulative distribution has been compared with results obtained in other parts of the world. The conversion factors Ce and CT obtained at Ile-Ife are generally lower than the values obtained for various locations in Europe. The number of occasions N(R) when a specified rainfall rate was continuously exceeded for durations of Δt seconds was found to obey the relationship N(R) = α(Δt)β, where α and β are functions of rainfall rate. At high rain rates, the majority of the return periods have been found to be due to rain rate variations within single cells and small mesoscale systems.

**IN “D. L. EMILIANI., L. LUINI, AND C. CAPSONI, “ANALYSIS AND PARAMETERIZATION OF METHODOLOGIES FOR THE CONVERSION OF RAIN RATE CUMULATIVE DISTRIBUTIONS FROM VARIOUS INTEGRATION TIMES TO ONE MINUTE,” IEEE ANTENNAS AND PROPAGATION MAGAZINE, VOL.51, NO.3, PP. 70-84, 2009”** The conversion of rain-rate cumulative distributions from any integration time, T, to one minute is a viable option whenever local one-minute data (time series or cumulative distribution functions) are not available for microwave system design. This paper reviews some of the most common rain-rate cumulative-distribution conversion methods. For selected models, it provides a complementary set of coefficients for regional and global application by performing regression to a measurements database. The performance of each model is analyzed, together with its adaptability to various climatic regions. Finally, recommendations with regard to the global applicability of models are given. The design and operation of microwave communication systems at frequencies above 10 GHz is heavily affected by tropospheric-propagation impairments. Among the various performance-affecting phenomena of interest, rain-induced attenuation is the most relevant [1], as the magnitudes ofthe fades exceed tens of dB at frequencies in the Ka (20 GHz to 30 GHz), QN (40 GHz to 50 GHz), or EHF (20 GHz to 45 GHz) bands. These bands are in use for both terrestrial services - such as wireless broadband access (BWA), local multipoint distribution service (LMDS), and multipoint video distribution systems (MVDS) - as well as for satellite networks, part of the fixed and broadcast satellite services (FSS, BSS). The investigation of rain-induced attenuation at such frequencies is made relevant not only because the amount of bandwidth available in the Ka and EHF bands enables the provision of complex multimedia applications, but also because of the current congestion that the Ku band is experiencing. This scenario makes higher-frequency bands more attractive for the deployment of new systems. Examples of new communication systems foreseen to operate in the Ka and EHF bands are the IEEE 802.16 BWA standard (planned both in licensed and unlicensed portions of the spectrum, from 11 up to 66 GHz [2]), and the DVB family ofstandards for satellite communications (DVB-S2 [3] and DVB-RCS, [4]) System design at these frequencies can be approached from two perspectives: A "traditional" approach, for which first-order statistics ofrelevant propagation phenomena (rain, clouds, gases, scintillation) are used in an iterative process. The objective is to guarantee that the received signal quality remains above a given threshold, having as constraints a quality and an availability objective expressed as percentage of a month or average year, as per lTV recommendations (such as [5-8]). In this case, first-order statistics ofrain playa key role. The accuracy of the models employed to estimate attenuation and rain rate exceeded for a given percentage of time at a given location is of the utmost importance. Various models have been proposed to estimate both parameters, but there is still uncertainty regarding the use of some propagation models of empirical nature outside their original domains [9-12] A "network-wide" approach, in which the incorporation of fade-mitigation techniques (FMTs) requires a change of design paradigm. This is because second-order statistics of fade dynamics (slope, duration), and issues such as the reaction time of the countermeasure (on a station-by-station basis) [13] and the overall permanence of the entire network (or area of coverage in a satellite system) in a given fade state (which dictates the amount of power and bandwidth needed on a system scale), are more relevant to guaranteeing the overall availability objective, still dictated by lTD recommendations. In this scenario, and as shown in [14, 15], firstorder statistics of rain and attenuation continue to be necessary, as they represent the inputs to the synthetic rain and attenuation models that are used to evaluate the network performance and to fine-tune the control loops in fade-mitigation techniques. This study deals with the estimation offirst-order statistics of rain rate required for the design of microwave communication systems. Specifically, it deals with the conversion of statistics of rain rate from an integration time, T, to the lTD-recommended value of one minute. This interval is recommended because it permits measuring the rain-rate temporal variation with adequate accuracy. Considering that the number of data sources fulfilling this condition is relatively small, the conversion of a cumulative distribution function} (CDF), P(R)r' with a longer integration time, T (typically ranging from five to 60 minutes), to a one-minute integrated CDF, P(R)t' through methods such as those proposed in [16, 17] represents a valid alternative. This latter solution should be preferred whenever local measurements are available, as it allows the preservation ofthe peculiarities ofthe rainfall process as much as possible, and thus improves the CDF-estimation accuracy [18].

## 1) Elucidating The Role Of Topological Pattern Discovery And Support Vector Machine In Generating Predictive Models For Indian Summer Monsoon Rainfall

**AUTHORS: Manojit Chattopadhyay, Surajit Chattopadhyay**

The present paper reports a study, where growing hierarchical self-organising map (GHSOM) has been applied to achieve a visual cluster analysis to the Indian rainfall dataset consisting of 142 years of Indian rainfall data so that the yearly rainfall can be segregated into small groups to visualise the pattern of clustering behaviour of yearly rainfall due to changes in monthly rainfall for each year. Also, through support vector machine (SVM), it has been observed that generation of clusters impacts positively on the prediction of the Indian summer monsoon rainfall. Results have been presented through statistical and graphical analyses. Behaviour of systems with many interdependent components that lead to organized as well as irregular features is referred to as complexity. In such systems the knowledge of the parts does not necessarily lead to the predictable behaviour of the entire system. Complexities associated with meteorological and geophysics processes have been reviewed in Sharma et al (2012). Modelling complexity of atmospheric phenomena and generating prediction schemes accordingly has long been an area of major concentration for the meteorologists over the globe (Kondratyev and Varotsos, 1995; Varotsos 2005, 2013, Blackwell, 2014). In view of importance of the estimation of the future projected precipitation and rainfall on short- and long-term basis detrended fluctuation analysis has been implemented by Efstathiou and Varotsos (2012) in rainfall time series to explore the intrinsic properties of their temporal variability. In another recent study, Chattopadhyay and Chattopadhyay (2013) explored the association between solar activity and Indian summer monsoon rainfall through spectral analysis after carrying out Box-Cox transformation. Association between SST and ENSO over the tropics has been discussed in a recent study by Varotsos et al. (2014), where they suggested that the warming in the sea surface temperature (SST) since 1900, did not occur smoothly and slowly, but with two rapid shifts in 1925/1926 and 1987/1988, which are more obvious over the tropics and the northern midlatitudes.

# 2) A Rainfall Prediction Model using Artificial Neural Network

**AUTHORS:**  **Kumar Abhishek, Abhay Kumar, Rajeev Ranjan, Sarthak Kumar**

The multilayered artificial neural network with learning by back-propagation algorithm configuration is the most common in use, due to of its ease in training. It is estimated that over 80% of all the neural network projects in development use back-propagation. In back-propagation algorithm, there are two phases in its learning cycle, one to propagate the input patterns through the network and other to adapt the output by changing the weights in the network. The back-propagation-feed forward neural network can be used in many applications such as character recognition, weather and financial prediction, face detection etc. The paper implements one of these applications by building training and testing data sets and finding the number of hidden neurons in these layers for the best performance. In the present research, possibility of predicting average rainfall over Udupi district of Karnataka has been analyzed through artificial neural network models. In formulating artificial neural network based predictive models three layered network has been constructed. The models under study are different in the number of hidden neurons.

# 3) A Short-Term Rainfall Prediction Model using Multi-Task Convolutional Neural Networks

**AUTHORS** **: Minghui Qiu, Peilin Zhao, Ke Zhang, Jun Huang, Xing Shi,**

**Xiaoguang Wang, Wei Chu**

# Precipitation prediction, such as short-term rainfall prediction, is a very important problem in the field of meteorological service. In practice, most of recent studies focus on leveraging radar data or satellite images to make predictions. However, there is another scenario where a set of weather features are collected by various sensors at multiple observation sites. The observations of a site are sometimes incomplete but provide important clues for weather prediction at nearby sites, which are not fully exploited in existing work yet. To solve this problem, we propose a multi-task convolutional neural network model to automatically extract features from the time series measured at observation sites and leverage the correlation between the multiple sites for weather prediction via multi-tasking. To the best of our knowledge, this is the first attempt to use multi-task learning and deep learning techniques to predict short-term rainfall amount based on multi-site features. Specifically, we formulate the learning task as an end-to-end multi-site neural network model which allows to leverage the learned knowledge from one site to other correlated sites, and model the correlations between different sites. Extensive experiments show that the learned site correlations are insightful and the proposed model significantly outperforms a broad set of baseline models including the European Centre for Medium-range Weather Forecasts system (ECMWF).

# 4) Deep Learning Models for the Prediction of Rainfall

**AUTHORS** : **Aswin S, Geetha P and Vinayakumar R**

Rainfall is one of the major source of freshwater for all the organism around the world. Rainfall prediction model provides the information regarding various climatological variables on the amount of rainfall. In recent days, Deep Learning enabled the self-learning data labels which allows to create a data-driven model for a time series dataset. It allows to make the anomaly/change detection from the time series data and also predicts the future event's data with respect to the events occurred in the past. This paper deals with obtaining models of the rainfall precipitation by using Deep Learning Architectures (LSTM and ConvNet) and determining the better architecture with RMSE of LSTM as 2.55 and RMSE of ConvNet as 2.44 claiming that for any time series dataset, Deep Learning models will be effective and efficient for the modellers.

**5)** **The Research Of Rainfall Prediction Models Based On Matlab Neural Network**

**AUTHORS**: Xianggen Gan, Lihong Chen, Dongbao Yang, Guang Liu

The continuously cloudy or rainy forecast is an important basis that is used to make choice of wheat harvest time but multiple regression weather forecast models hardly content the rate of required accuracy. Matlab neural network toolbox is composed of a series of typical neural network activation functions that make computing network output into calling activation functions. BP artificial neural network that is based on Matlab platform and utilizes error back propagation algorithm to revise network weight has dynamic frame characteristics and is convenient for constructing network and programming. After it has been trained by input forecast samples, network forecast model that has three neural cells possesses very good generalization capability. After we contrast fitting rate and accuracy rate of network model with ones of regression model, network model has a distinct advantage over regression model.

**PROPOSED SYSTEM:**

A neural network is a distributed processor that consists of artificial neurons as primary processing elements. Neural networks can be used for many applications including pattern classification, function approximation, clustering, prediction/forecasting, optimization, content addressable memory. Rainfall, being a highly non-linear phenomenon, requires a more complex non-statistical method for its prediction, such as an artificial neural network . An artificial neural network can be trained either via supervised or unsupervised learning. In the former method, the network is presented with a set of inputs and their corresponding desired output(s), also referred to as targets, for each iteration. Thereafter, the outputs are compared with targets for determination of the magnitude of errors that are then used in the adjustment of the network weights in the negative direction (gradient descent). On the other hand, in the unsupervised learning, the neural network is capable of drawing inferences from a dataset consisting of only inputs with no targets. This paper presents a predictive model approach that focuses on developing a prediction technique that can predict the rainfall rate ahead of time. Attenuation resulting from the predicted rainfall rate is then used to select an appropriate digital modulation technique that will ensure availability of the link and the quality of service being offered.

**ADVANTAGES:**

• Weather Prediction time consequence is very fast.

• Prediction result is very accurate and clearly process.

• Dataset analysis and management is high level process and image pre-Processing one of best analysis process in weather prediction system.

**REQUIREMENT ANALYSIS**

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

**REQUIREMENT SPECIFICATION**

**Functional Requirements**

* Graphical User interface with the User.

**Software Requirements**

For developing the application the following are the Software Requirements:

1. Python
2. Django

**Operating Systems supported**

1. Windows 7
2. Windows XP
3. Windows 8

**Technologies and Languages used to Develop**

1. Python

**Debugger and Emulator**

* Any Browser (Particularly Chrome)

**Hardware Requirements**

For developing the application the following are the Hardware Requirements:

* Processor: Pentium IV or higher
* RAM: 256 MB
* Space on Hard Disk: minimum 512MB

**INPUT AND OUTPUT DESIGN**

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**SYSTEM STUDY**

**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**Three key considerations involved in the feasibility analysis are,**

* **ECONOMICAL FEASIBILITY**
* **TECHNICAL FEASIBILITY**
* **SOCIAL FEASIBILITY**

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**SYSTEM SPECIFICATION:**

**HARDWARE REQUIREMENTS:**

* **System :** Pentium IV 2.4 GHz.
* **Hard Disk :** 40 GB.
* **Floppy Drive :** 1.44 Mb.
* **Monitor** : 14’ Colour Monitor.
* **Mouse :** Optical Mouse.
* **Ram :** 512 Mb.

**SOFTWARE REQUIREMENTS:**

* **Operating system :** Windows 7 Ultimate.
* **Coding Language :** Python.
* **Front-End :** Python.
* **Designing :** Html,css,javascript.
* **Data Base :** MySQL.

**PYTHON**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An [interpreted language](https://en.wikipedia.org/wiki/Interpreted_language), Python has a design philosophy that emphasizes code [readability](https://en.wikipedia.org/wiki/Readability) (notably using [whitespace](https://en.wikipedia.org/wiki/Whitespace_character) indentation to delimit [code blocks](https://en.wikipedia.org/wiki/Code_block) rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer [lines of code](https://en.wikipedia.org/wiki/Source_lines_of_code) than might be used in languages such as [C++](https://en.wikipedia.org/wiki/C%2B%2B)or [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [CPython](https://en.wikipedia.org/wiki/CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open source](https://en.wikipedia.org/wiki/Open_source) software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation). Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming), and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library)

**DJANGO**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes [reusability](https://en.wikipedia.org/wiki/Reusability)and "pluggability" of components, rapid development, and the principle of [don't repeat yourself](https://en.wikipedia.org/wiki/Don%27t_repeat_yourself). Python is used throughout, even for settings files and data models.



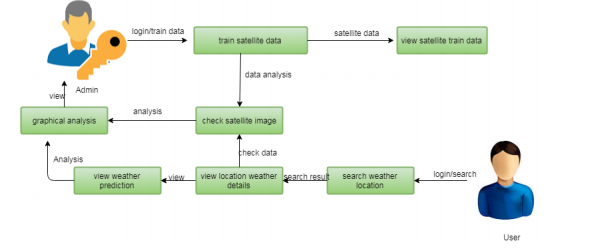
Django also provides an optional administrative [create, read, update and delete](https://en.wikipedia.org/wiki/Create,_read,_update_and_delete) interface that is generated dynamically through [introspection](https://en.wikipedia.org/wiki/Introspection_(computer_science)) and configured via admin models



**IMPLEMENTATION**

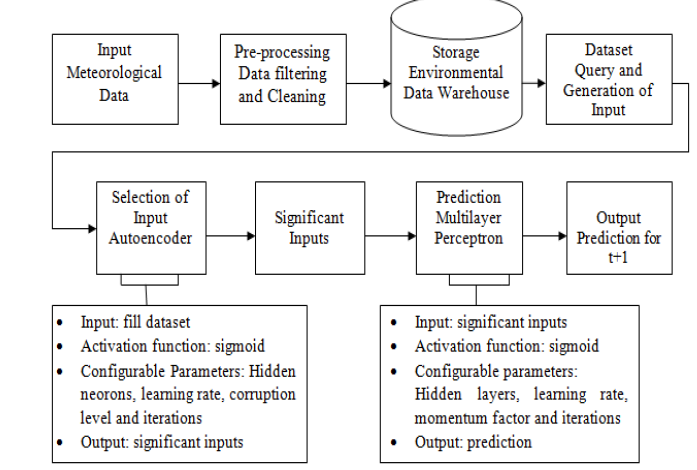
This system consists of two users admin and user. The data can be uploaded by admin without any particular scenario but with the details of satellite data. The most importantly large amount of can be handled in order to do practically. The data that are handling throughout the project can be done in this module. Users have permission to view data but not edit the data in online they can request the user to get the data

**SYSTEM ARCHITECTIURE:**



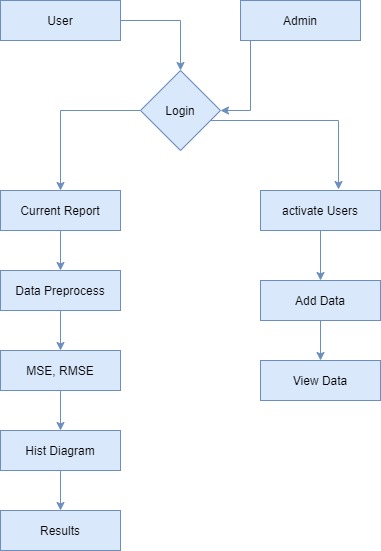
**SYSTEM DESIGN**

**SYSTEM ARCHITECTURE:**



**DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. 

**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



**SYSTEM TEST**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### TYPES OF TESTS

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Sample Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.no** | **Test Case** | **Excepted Result** | **Result** | **Remarks(IF Fails)** |
| 1. | User Register | If User registration successfully. | Pass | If already user email exist then it fails. |
| 2. | User Login | If User name and password is correct then it will getting valid page. | Pass | Un Register Users will not logged in. |
| 3. | Admin Add the Data | A new record will added to our dataset. | Pass | According to India metrological repository the data must be float fields otherwise its failed. |
| 4. | Data Cleaning | Data will be cleaned. | Pass | The data will be in int or float format, otherwise algorithm will not work.. |
| 5. | Hist digram | Hist diagram generated bsed on months | Pass | If data not cleaned or NA then it give you an error |
| 6. | Admin dd extra records for testing | Data will be consider for testing purpose. | Pass | Data added to test data for model. |
| 7. | Calculate MSE | Means square error calculated and displayed | Pass | Data is consider for testing. |
| 8. | Root Mean Square Error calculation | RootMeans Square calculated | Pass | Root Meansquare calculated |
| 9. | Admin login | Admin can login with his login credential. If success he get his home page | Pass | Invalid login details will not allowed here |
| 10. | Admin can activate the register users | Admin can activate the register user id | Pass | If user id not found then it won’t login. |

**ALGORITHMS:**

**ANN:**

Artificial neural networks (ANNs), usually simply called neural networks (NNs), are computing systems vaguely inspired by the biological neural networks that constitute animal brains. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times. Neural networks learn (or are trained) by processing examples, each of which contains a known "input" and "result," forming probability-weighted associations between the two, which are stored within the data structure of the net itself. The training of a neural network from a given example is usually conducted by determining the difference between the processed output of the network (often a prediction) and a target output. This is the error. The network then adjusts its weighted associations according to a learning rule and using this error value. Successive adjustments will cause the neural network to produce output which is increasingly similar to the target output. After a sufficient number of these adjustments the training can be terminated based upon certain criteria. This is known as supervised learning.

Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that they have fur, tails, whiskers, and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process.

History

Main article: History of artificial neural networks

Warren McCulloch and Walter Pitts[1] (1943) opened the subject by creating a computational model for neural networks.[2] In the late 1940s, D. O. Hebb[3] created a learning hypothesis based on the mechanism of neural plasticity that became known as Hebbian learning. Farley and Wesley A. Clark[4] (1954) first used computational machines, then called "calculators", to simulate a Hebbian network. Rosenblatt[5] (1958) created the perceptron.[6] The first functional networks with many layers were published by Ivakhnenko and Lapa in 1965, as the Group Method of Data Handling.[7][8][9] The basics of continuous backpropagation[7][10][11][12] were derived in the context of control theory by Kelley[13] in 1960 and by Bryson in 1961,[14] using principles of dynamic programming.

In 1970, Seppo Linnainmaa published the general method for automatic differentiation (AD) of discrete connected networks of nested differentiable functions.[15][16] In 1973, Dreyfus used backpropagation to adapt parameters of controllers in proportion to error gradients.[17] Werbos's (1975) backpropagation algorithm enabled practical training of multi-layer networks. In 1982, he applied Linnainmaa's AD method to neural networks in the way that became widely used.[10][18] Thereafter research stagnated following Minsky and Papert (1969),[19] who discovered that basic perceptrons were incapable of processing the exclusive-or circuit and that computers lacked sufficient power to process useful neural networks.

The development of metal–oxide–semiconductor (MOS) very-large-scale integration (VLSI), in the form of complementary MOS (CMOS) technology, enabled increasing MOS transistor counts in digital electronics. This provided more processing power for the development of practical artificial neural networks in the 1980s.[20]

In 1986 Rumelhart, Hinton and Williams showed that backpropagation learned interesting internal representations of words as feature vectors when trained to predict the next word in a sequence.[21]

In 1992, max-pooling was introduced to help with least-shift invariance and tolerance to deformation to aid 3D object recognition.[22][23][24] Schmidhuber adopted a multi-level hierarchy of networks (1992) pre-trained one level at a time by unsupervised learning and fine-tuned by backpropagation.[25]

Geoffrey Hinton et al. (2006) proposed learning a high-level representation using successive layers of binary or real-valued latent variables with a restricted Boltzmann machine[26] to model each layer. In 2012, Ng and Dean created a network that learned to recognize higher-level concepts, such as cats, only from watching unlabeled images.[27] Unsupervised pre-training and increased computing power from GPUs and distributed computing allowed the use of larger networks, particularly in image and visual recognition problems, which became known as "deep learning".[28]

Ciresan and colleagues (2010)[29] showed that despite the vanishing gradient problem, GPUs make backpropagation feasible for many-layered feedforward neural networks.[30] Between 2009 and 2012, ANNs began winning prizes in ANN contests, approaching human level performance on various tasks, initially in pattern recognition and machine learning.[31][32] For example, the bi-directional and multi-dimensional long short-term memory (LSTM)[33][34][35][36] of Graves et al. won three competitions in connected handwriting recognition in 2009 without any prior knowledge about the three languages to be learned.[35][34]

Ciresan and colleagues built the first pattern recognizers to achieve human-competitive/superhuman performance[37] on benchmarks such as traffic sign recognition (IJCNN 2012).

Models

This section may be confusing or unclear to readers. Please help clarify the section. There might be a discussion about this on the talk page. (April 2017) (Learn how and when to remove this template message)

Further information: Mathematics of artificial neural networks

Neuron and myelinated axon, with signal flow from inputs at dendrites to outputs at axon terminals

ANNs began as an attempt to exploit the architecture of the human brain to perform tasks that conventional algorithms had little success with. They soon reoriented towards improving empirical results, mostly abandoning attempts to remain true to their biological precursors. Neurons are connected to each other in various patterns, to allow the output of some neurons to become the input of others. The network forms a directed, weighted graph.[38]

An artificial neural network consists of a collection of simulated neurons. Each neuron is a node which is connected to other nodes via links that correspond to biological axon-synapse-dendrite connections. Each link has a weight, which determines the strength of one node's influence on another.[39]

Components of ANNs

Neurons

ANNs are composed of artificial neurons which are conceptually derived from biological neurons. Each artificial neuron has inputs and produces a single output which can be sent to multiple other neurons. The inputs can be the feature values of a sample of external data, such as images or documents, or they can be the outputs of other neurons. The outputs of the final output neurons of the neural net accomplish the task, such as recognizing an object in an image.

To find the output of the neuron, first we take the weighted sum of all the inputs, weighted by the weights of the connections from the inputs to the neuron. We add a bias term to this sum. This weighted sum is sometimes called the activation. This weighted sum is then passed through a (usually nonlinear) activation function to produce the output. The initial inputs are external data, such as images and documents. The ultimate outputs accomplish the task, such as recognizing an object in an image.[40]

Connections and weights

The network consists of connections, each connection providing the output of one neuron as an input to another neuron. Each connection is assigned a weight that represents its relative importance.[38] A given neuron can have multiple input and output connections.[41]

Propagation function

The propagation function computes the input to a neuron from the outputs of its predecessor neurons and their connections as a weighted sum.[38] A bias term can be added to the result of the propagation.[42]

Organization

The neurons are typically organized into multiple layers, especially in deep learning. Neurons of one layer connect only to neurons of the immediately preceding and immediately following layers. The layer that receives external data is the input layer. The layer that produces the ultimate result is the output layer. In between them are zero or more hidden layers. Single layer and unlayered networks are also used. Between two layers, multiple connection patterns are possible. They can be fully connected, with every neuron in one layer connecting to every neuron in the next layer. They can be pooling, where a group of neurons in one layer connect to a single neuron in the next layer, thereby reducing the number of neurons in that layer.[43] Neurons with only such connections form a directed acyclic graph and are known as feedforward networks.[44] Alternatively, networks that allow connections between neurons in the same or previous layers are known as recurrent networks.[45]

Hyperparameter

Main article: Hyperparameter (machine learning)

A hyperparameter is a constant parameter whose value is set before the learning process begins. The values of parameters are derived via learning. Examples of hyperparameters include learning rate, the number of hidden layers and batch size.[46] The values of some hyperparameters can be dependent on those of other hyperparameters. For example, the size of some layers can depend on the overall number of layers.

Learning

This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message)

See also: Mathematical optimization, Estimation theory, and Machine learning

Learning is the adaptation of the network to better handle a task by considering sample observations. Learning involves adjusting the weights (and optional thresholds) of the network to improve the accuracy of the result. This is done by minimizing the observed errors. Learning is complete when examining additional observations does not usefully reduce the error rate. Even after learning, the error rate typically does not reach 0. If after learning, the error rate is too high, the network typically must be redesigned. Practically this is done by defining a cost function that is evaluated periodically during learning. As long as its output continues to decline, learning continues. The cost is frequently defined as a statistic whose value can only be approximated. The outputs are actually numbers, so when the error is low, the difference between the output (almost certainly a cat) and the correct answer (cat) is small. Learning attempts to reduce the total of the differences across the observations.[38] Most learning models can be viewed as a straightforward application of optimization theory and statistical estimation.

Learning rate

The learning rate defines the size of the corrective steps that the model takes to adjust for errors in each observation. A high learning rate shortens the training time, but with lower ultimate accuracy, while a lower learning rate takes longer, but with the potential for greater accuracy. Optimizations such as Quickprop are primarily aimed at speeding up error minimization, while other improvements mainly try to increase reliability. In order to avoid oscillation inside the network such as alternating connection weights, and to improve the rate of convergence, refinements use an adaptive learning rate that increases or decreases as appropriate.[47] The concept of momentum allows the balance between the gradient and the previous change to be weighted such that the weight adjustment depends to some degree on the previous change. A momentum close to 0 emphasizes the gradient, while a value close to 1 emphasizes the last change.

Cost function

While it is possible to define a cost function ad hoc, frequently the choice is determined by the function's desirable properties (such as convexity) or because it arises from the model (e.g. in a probabilistic model the model's posterior probability can be used as an inverse cost).

Backpropagation

Main article: Backpropagation

Backpropagation is a method used to adjust the connection weights to compensate for each error found during learning. The error amount is effectively divided among the connections. Technically, backprop calculates the gradient (the derivative) of the cost function associated with a given state with respect to the weights. The weight updates can be done via stochastic gradient descent or other methods, such as Extreme Learning Machines,[48] "No-prop" networks,[49] training without backtracking,[50] "weightless" networks,[51][52] and non-connectionist neural networks.

Learning paradigms

This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message)

The three major learning paradigms are supervised learning, unsupervised learning and reinforcement learning. They each correspond to a particular learning task

Supervised learning

Supervised learning uses a set of paired inputs and desired outputs. The learning task is to produce the desired output for each input. In this case the cost function is related to eliminating incorrect deductions.[53] A commonly used cost is the mean-squared error, which tries to minimize the average squared error between the network's output and the desired output. Tasks suited for supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). Supervised learning is also applicable to sequential data (e.g., for hand writing, speech and gesture recognition). This can be thought of as learning with a "teacher", in the form of a function that provides continuous feedback on the quality of solutions obtained thus far.

Unsupervised learning

In unsupervised learning, input data is given along with the cost function, some function of the data {\displaystyle \textstyle x}\textstyle x and the network's output. The cost function is dependent on the task (the model domain) and any a priori assumptions (the implicit properties of the model, its parameters and the observed variables). As a trivial example, consider the model {\displaystyle \textstyle f(x)=a}\textstyle f(x)=a where {\displaystyle \textstyle a}\textstyle a is a constant and the cost {\displaystyle \textstyle C=E[(x-f(x))^{2}]}\textstyle C=E[(x-f(x))^{2}]. Minimizing this cost produces a value of {\displaystyle \textstyle a}\textstyle a that is equal to the mean of the data. The cost function can be much more complicated. Its form depends on the application: for example, in compression it could be related to the mutual information between {\displaystyle \textstyle x}\textstyle x and {\displaystyle \textstyle f(x)}\textstyle f(x), whereas in statistical modeling, it could be related to the posterior probability of the model given the data (note that in both of those examples those quantities would be maximized rather than minimized). Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering.

Reinforcement learning

Main article: Reinforcement learning

See also: Stochastic control

In applications such as playing video games, an actor takes a string of actions, receiving a generally unpredictable response from the environment after each one. The goal is to win the game, i.e., generate the most positive (lowest cost) responses. In reinforcement learning, the aim is to weight the network (devise a policy) to perform actions that minimize long-term (expected cumulative) cost. At each point in time the agent performs an action and the environment generates an observation and an instantaneous cost, according to some (usually unknown) rules. The rules and the long-term cost usually only can be estimated. At any juncture, the agent decides whether to explore new actions to uncover their costs or to exploit prior learning to proceed more quickly.

Formally the environment is modeled as a Markov decision process (MDP) with states {\displaystyle \textstyle {s\_{1},...,s\_{n}}\in S}\textstyle {s\_{1},...,s\_{n}}\in S and actions {\displaystyle \textstyle {a\_{1},...,a\_{m}}\in A}\textstyle {a\_{1},...,a\_{m}}\in A. Because the state transitions are not known, probability distributions are used instead: the instantaneous cost distribution {\displaystyle \textstyle P(c\_{t}|s\_{t})}\textstyle P(c\_{t}|s\_{t}), the observation distribution {\displaystyle \textstyle P(x\_{t}|s\_{t})}\textstyle P(x\_{t}|s\_{t}) and the transition distribution {\displaystyle \textstyle P(s\_{t+1}|s\_{t},a\_{t})}\textstyle P(s\_{t+1}|s\_{t},a\_{t}), while a policy is defined as the conditional distribution over actions given the observations. Taken together, the two define a Markov chain (MC). The aim is to discover the lowest-cost MC.

ANNs serve as the learning component in such applications.[54][55] Dynamic programming coupled with ANNs (giving neurodynamic programming)[56] has been applied to problems such as those involved in vehicle routing,[57] video games, natural resource management[58][59] and medicine[60] because of ANNs ability to mitigate losses of accuracy even when reducing the discretization grid density for numerically approximating the solution of control problems. Tasks that fall within the paradigm of reinforcement learning are control problems, games and other sequential decision making tasks.

Self learning

Self learning in neural networks was introduced in 1982 along with a neural network capable of self-learning named Crossbar Adaptive Array (CAA).[61] It is a system with only one input, situation s, and only one output, action (or behavior) a. It has neither external advice input nor external reinforcement input from the environment. The CAA computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about encountered situations. The system is driven by the interaction between cognition and emotion.[62] Given memory matrix W =||w(a,s)||, the crossbar self learning algorithm in each iteration performs the following computation:

In situation s perform action a;

Receive consequence situation s';

Compute emotion of being in consequence situation v(s');

Update crossbar memory w'(a,s) = w(a,s) + v(s').

The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is behavioral environment where it behaves, and the other is genetic environment, where from it initially and only once receives initial emotions about to be encountered situations in the behavioral environment. Having received the genome vector (species vector) from the genetic environment, the CAA will learn a goal-seeking behavior, in the behavioral environment that contains both desirable and undesirable situations.[63]

Other

In a Bayesian framework, a distribution over the set of allowed models is chosen to minimize the cost. Evolutionary methods,[64] gene expression programming,[65] simulated annealing,[66] expectation-maximization, non-parametric methods and particle swarm optimization[67] are other learning algorithms. Convergent recursion is a learning algorithm for cerebellar model articulation controller (CMAC) neural networks.[68][69]

Modes

This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message)

Two modes of learning are available: stochastic and batch. In stochastic learning, each input creates a weight adjustment. In batch learning weights are adjusted based on a batch of inputs, accumulating errors over the batch. Stochastic learning introduces "noise" into the process, using the local gradient calculated from one data point; this reduces the chance of the network getting stuck in local minima. However, batch learning typically yields a faster, more stable descent to a local minimum, since each update is performed in the direction of the batch's average error. A common compromise is to use "mini-batches", small batches with samples in each batch selected stochastically from the entire data set.

Types

Main article: Types of artificial neural networks

ANNs have evolved into a broad family of techniques that have advanced the state of the art across multiple domains. The simplest types have one or more static components, including number of units, number of layers, unit weights and topology. Dynamic types allow one or more of these to evolve via learning. The latter are much more complicated, but can shorten learning periods and produce better results. Some types allow/require learning to be "supervised" by the operator, while others operate independently. Some types operate purely in hardware, while others are purely software and run on general purpose computers.

Some of the main breakthroughs include: convolutional neural networks that have proven particularly successful in processing visual and other two-dimensional data;[70][71] long short-term memory avoid the vanishing gradient problem[72] and can handle signals that have a mix of low and high frequency components aiding large-vocabulary speech recognition,[73][74] text-to-speech synthesis,[75][10][76] and photo-real talking heads;[77] competitive networks such as generative adversarial networks in which multiple networks (of varying structure) compete with each other, on tasks such as winning a game[78] or on deceiving the opponent about the authenticity of an input.[79]

Network design

Main article: Neural architecture search

Neural architecture search (NAS) uses machine learning to automate ANN design. Various approaches to NAS have designed networks that compare well with hand-designed systems. The basic search algorithm is to propose a candidate model, evaluate it against a dataset and use the results as feedback to teach the NAS network.[80] Available systems include AutoML and AutoKeras.[81]

Design issues include deciding the number, type and connectedness of network layers, as well as the size of each and the connection type (full, pooling, ...).

Hyperparameters must also be defined as part of the design (they are not learned), governing matters such as how many neurons are in each layer, learning rate, step, stride, depth, receptive field and padding (for CNNs), etc.[82]

Use

This section does not cite any sources. Please help improve this section by adding citations to reliable sources. Unsourced material may be challenged and removed. (November 2020) (Learn how and when to remove this template message)

Using Artificial neural networks requires an understanding of their characteristics.

Choice of model: This depends on the data representation and the application. Overly complex models slow learning.

Learning algorithm: Numerous trade-offs exist between learning algorithms. Almost any algorithm will work well with the correct hyperparameters for training on a particular data set. However, selecting and tuning an algorithm for training on unseen data requires significant experimentation.

Robustness: If the model, cost function and learning algorithm are selected appropriately, the resulting ANN can become robust.

ANN capabilities fall within the following broad categories:[citation needed]

Function approximation, or regression analysis, including time series prediction, fitness approximation and modeling.

Classification, including pattern and sequence recognition, novelty detection and sequential decision making.[83]

Data processing, including filtering, clustering, blind source separation and compression.

Robotics, including directing manipulators and prostheses.

Applications

Because of their ability to reproduce and model nonlinear processes, Artificial neural networks have found applications in many disciplines. Application areas include system identification and control (vehicle control, trajectory prediction,[84] process control, natural resource management), quantum chemistry,[85] general game playing,[86] pattern recognition (radar systems, face identification, signal classification,[87] 3D reconstruction,[88] object recognition and more), sequence recognition (gesture, speech, handwritten and printed text recognition[89]), medical diagnosis, finance[90] (e.g. automated trading systems), data mining, visualization, machine translation, social network filtering[91] and e-mail spam filtering. ANNs have been used to diagnose several types of cancers[92][93] and to distinguish highly invasive cancer cell lines from less invasive lines using only cell shape information.[94][95]

ANNs have been used to accelerate reliability analysis of infrastructures subject to natural disasters[96][97] and to predict foundation settlements.[98] ANNs have also been used for building black-box models in geoscience: hydrology,[99][100] ocean modelling and coastal engineering,[101][102] and geomorphology.[103] ANNs have been employed in cybersecurity, with the objective to discriminate between legitimate activities and malicious ones. For example, machine learning has been used for classifying Android malware,[104] for identifying domains belonging to threat actors and for detecting URLs posing a security risk.[105] Research is underway on ANN systems designed for penetration testing, for detecting botnets,[106] credit cards frauds[107] and network intrusions.

ANNs have been proposed as a tool to solve partial differential equations in physics[108][109] and simulate the properties of many-body open quantum systems.[110][111][112][113] In brain research ANNs have studied short-term behavior of individual neurons,[114] the dynamics of neural circuitry arise from interactions between individual neurons and how behavior can arise from abstract neural modules that represent complete subsystems. Studies considered long-and short-term plasticity of neural systems and their relation to learning and memory from the individual neuron to the system level.

Theoretical properties

Computational power

The multilayer perceptron is a universal function approximator, as proven by the universal approximation theorem. However, the proof is not constructive regarding the number of neurons required, the network topology, the weights and the learning parameters.

A specific recurrent architecture with rational-valued weights (as opposed to full precision real number-valued weights) has the power of a universal Turing machine,[115] using a finite number of neurons and standard linear connections. Further, the use of irrational values for weights results in a machine with super-Turing power.[116]

Capacity

A model's "capacity" property corresponds to its ability to model any given function. It is related to the amount of information that can be stored in the network and to the notion of complexity. Two notions of capacity are known by the community. The information capacity and the VC Dimension. The information capacity of a perceptron is intensively discussed in Sir David MacKay's book[117] which summarizes work by Thomas Cover.[118] The capacity of a network of standard neurons (not convolutional) can be derived by four rules[119] that derive from understanding a neuron as an electrical element. The information capacity captures the functions modelable by the network given any data as input. The second notion, is the VC dimension. VC Dimension uses the principles of measure theory and finds the maximum capacity under the best possible circumstances. This is, given input data in a specific form. As noted in,[117] the VC Dimension for arbitrary inputs is half the information capacity of a Perceptron. The VC Dimension for arbitrary points is sometimes referred to as Memory Capacity.[120]

Convergence

Models may not consistently converge on a single solution, firstly because local minima may exist, depending on the cost function and the model. Secondly, the optimization method used might not guarantee to converge when it begins far from any local minimum. Thirdly, for sufficiently large data or parameters, some methods become impractical.

The convergence behavior of certain types of ANN architectures are more understood than others. When the width of network approaches to infinity, the ANN is well described by its first order Taylor expansion throughout training, and so inherits the convergence behavior of affine models.[121][122] Another example is when parameters are small, it is observed that ANNs often fits target functions from low to high frequencies.[123][124][125][126] This phenomenon is the opposite to the behavior of some well studied iterative numerical schemes such as Jacobi method.

Generalization and statistics

This section includes a list of references, related reading or external links, but its sources remain unclear because it lacks inline citations. Please help to improve this section by introducing more precise citations. (August 2019) (Learn how and when to remove this template message)

Applications whose goal is to create a system that generalizes well to unseen examples, face the possibility of over-training. This arises in convoluted or over-specified systems when the network capacity significantly exceeds the needed free parameters. Two approaches address over-training. The first is to use cross-validation and similar techniques to check for the presence of over-training and to select hyperparameters to minimize the generalization error.

The second is to use some form of regularization. This concept emerges in a probabilistic (Bayesian) framework, where regularization can be performed by selecting a larger prior probability over simpler models; but also in statistical learning theory, where the goal is to minimize over two quantities: the 'empirical risk' and the 'structural risk', which roughly corresponds to the error over the training set and the predicted error in unseen data due to overfitting.

Confidence analysis of a neural network

Supervised neural networks that use a mean squared error (MSE) cost function can use formal statistical methods to determine the confidence of the trained model. The MSE on a validation set can be used as an estimate for variance. This value can then be used to calculate the confidence interval of network output, assuming a normal distribution. A confidence analysis made this way is statistically valid as long as the output probability distribution stays the same and the network is not modified.

By assigning a softmax activation function, a generalization of the logistic function, on the output layer of the neural network (or a softmax component in a component-based network) for categorical target variables, the outputs can be interpreted as posterior probabilities. This is useful in classification as it gives a certainty measure on classifications.

The softmax activation function is:

{\displaystyle y\_{i}={\frac {e^{x\_{i}}}{\sum \_{j=1}^{c}e^{x\_{j}}}}}y\_{i}={\frac {e^{x\_{i}}}{\sum \_{j=1}^{c}e^{x\_{j}}}}

Criticism

Training

A common criticism of neural networks, particularly in robotics, is that they require too much training for real-world operation.[citation needed] Potential solutions include randomly shuffling training examples, by using a numerical optimization algorithm that does not take too large steps when changing the network connections following an example, grouping examples in so-called mini-batches and/or introducing a recursive least squares algorithm for CMAC.[68]

Theory

A fundamental objection is that ANNs do not sufficiently reflect neuronal function. Backpropagation is a critical step, although no such mechanism exists in biological neural networks.[127] How information is coded by real neurons is not known. Sensor neurons fire action potentials more frequently with sensor activation and muscle cells pull more strongly when their associated motor neurons receive action potentials more frequently.[128] Other than the case of relaying information from a sensor neuron to a motor neuron, almost nothing of the principles of how information is handled by biological neural networks is known.

A central claim of ANNs is that they embody new and powerful general principles for processing information. These principles are ill-defined. It is often claimed that they are emergent from the network itself. This allows simple statistical association (the basic function of artificial neural networks) to be described as learning or recognition. Alexander Dewdney commented that, as a result, artificial neural networks have a "something-for-nothing quality, one that imparts a peculiar aura of laziness and a distinct lack of curiosity about just how good these computing systems are. No human hand (or mind) intervenes; solutions are found as if by magic; and no one, it seems, has learned anything".[129] One response to Dewdney is that neural networks handle many complex and diverse tasks, ranging from autonomously flying aircraft[130] to detecting credit card fraud to mastering the game of Go.

Technology writer Roger Bridgman commented:

Neural networks, for instance, are in the dock not only because they have been hyped to high heaven, (what hasn't?) but also because you could create a successful net without understanding how it worked: the bunch of numbers that captures its behaviour would in all probability be "an opaque, unreadable table...valueless as a scientific resource".

In spite of his emphatic declaration that science is not technology, Dewdney seems here to pillory neural nets as bad science when most of those devising them are just trying to be good engineers. An unreadable table that a useful machine could read would still be well worth having.[131]

Biological brains use both shallow and deep circuits as reported by brain anatomy,[132] displaying a wide variety of invariance. Weng[133] argued that the brain self-wires largely according to signal statistics and therefore, a serial cascade cannot catch all major statistical dependencies.

Hardware

Large and effective neural networks require considerable computing resources.[134] While the brain has hardware tailored to the task of processing signals through a graph of neurons, simulating even a simplified neuron on von Neumann architecture may consume vast amounts of memory and storage. Furthermore, the designer often needs to transmit signals through many of these connections and their associated neurons – which require enormous CPU power and time.

Schmidhuber noted that the resurgence of neural networks in the twenty-first century is largely attributable to advances in hardware: from 1991 to 2015, computing power, especially as delivered by GPGPUs (on GPUs), has increased around a million-fold, making the standard backpropagation algorithm feasible for training networks that are several layers deeper than before.[7] The use of accelerators such as FPGAs and GPUs can reduce training times from months to days.[135][134]

Neuromorphic engineering addresses the hardware difficulty directly, by constructing non-von-Neumann chips to directly implement neural networks in circuitry. Another type of chip optimized for neural network processing is called a Tensor Processing Unit, or TPU.[136]

Practical counterexamples

Analyzing what has been learned by an ANN is much easier than analyzing what has been learned by a biological neural network. Furthermore, researchers involved in exploring learning algorithms for neural networks are gradually uncovering general principles that allow a learning machine to be successful. For example, local vs. non-local learning and shallow vs. deep architecture.[137]

Hybrid approaches

Advocates of hybrid models (combining neural networks and symbolic approaches), claim that such a mixture can better capture the mechanisms of the human mind

CONCLUSION Using three past rainfall rates in the interval of 30 seconds from time ሺݐെʹሻ to time ሺݐሻ, the neural network based rainfall prediction model developed in this study was successful in predicting a rainfall rate 30 seconds ahead of time in a sliding window format. Error analysis using RMSE values as low as 0.1542 for a drizzle rainfall event were realized and this confirms that the backpropagation neural network can be trained and used to predict rainfall rates for estimation of rain fade attenuation on an earth-satellite link. The estimated attenuation can then be used in setting up a fade counter-measure in due time for provision of continuous reception of quality service on a microwave link. REFERENCES [1] T. S. Rappaport, “Millimeter Wave Wireless Communications: The Renaissance of Computing and Communications,” Internatinoal Conference on Communications, Australia, June, 2014. [2] G. O. Ajayi and E.B.C. Ofoche, “Some Tropical Rain Rate Characteristics at Ile-Ife for Microwave and Millimeter Wave Applications,” Journal of Climate and Applied Meteorology, Vol. 23, pp. 562-567, 1984. [3] D. L. Emiliani., L. Luini, and C. Capsoni, “Analysis and parameterization of methodologies for the conversion of rain rate cumulative distributions from various integration times to one minute,” IEEE Antennas and Propagation Magazine, Vol.51, No.3, pp. 70-84, 2009. [4] L. Dossi, G. Tartar and E. Matricciani, “Frequency Diversity in Millimeter Wave Satellite Communications,” IEEE Transactions on Aerospace and Electronic Systems, Vol. 28, No. 2, April 1992. [5] L. J. Ippolito, Jr., Satellite Communications Systems Engineering: Atmospheric Effects, Satellite Link Design and System Performance, John Wiley and Sons Ltd, UK, 2008. [6] N.W.M. Saad, A.F. Ismail, K. Badron and N.H.M Sobli, “Assessments Time Diversity Rain Fade Mitigation Technique for V-band Space-Earth Link Operating in Tropical Climate,” International Journal of Electrical Energy, Vol. 1, No.4, December, 2013. [7] F. A. Molisch, Wireless Communications. Second Ed., Wiley, 2011. [8] K. Abhishek, A. Kumar, R. Ranjan and S. Kumar, “A rainfall Prediction Model using Artificial Neural Network,” IEEE Control and System Graduate Research Colloquium, 2012. [9] G. M. N. French, W.F. Krajewski and R.R. Cuykendall, “Rainfall Forecasting in Space and time using neural network”, J. Hydrol. Vol. 137, pp. 1-31, 1992. [10] D. R. Nayak, A. Mahapatra and P. Mishra, “A Survey on Rainfall Prediction using Artificial Neural Network,” International Journal of Computer Applications, Vol. 72, No. 16, June 2013. [11] S. C. Michaelides, C. C. Neocleous & C. N. Schizas, “Artificial neural networks and multiple linear regressions in estimating missing rainfall data.” In: Proceedings of the DSP95 International Conference on Digital Signal Processing, Limassol, Cyprus. pp 668–673, 1995. [12] C. I., Christodoulou, S.C., M. Gabella and C.S. Pattichis, “Prediction of Rainfall Rate based on weather radar measurement,” IEEE, 2004 Vol. 2 (1393-1396). [13] R. Rojas, Neural Networks. A systematic Introduction. Springer-Verlag, Berlin, 1996. [14] D. E. Rumelhart, G.E. Hinton and R.J. Williams, “Learning Internal Representations by Error Propagation,” in Parallel Distributed Processing, Vol. 1, D.E. Rumelhart and J.L. McClelland, Editors, Cambridge, M.A.: MIT Prss, 1986, pp. 318-362. [15] Distromet system (2000), The Joss-Waldvogel Disdrometer Handbook, Basel, Switzerland [16] A. A. Alonge and Thomas J. Afullo, “Estimtion of Parameters for Lognormal Rainfall DSD Model for Various Rainfall Types in Durban,” SATNAC, 2011. [17] M. N. Ahuna, Thomas J. Afullo and Akintunde A. Alonge, “30-Second and One-Minute Rainfall Rate Modelling and Conversion for Millimetric Wave Propagation in South Africa,” SAIEE, Vol. 107 (1), March 2016, pp. 17-29. [18] M. Galoie, G. Zenz and A. Motamedi: “Rainfall analysis for the Schoeckelbach Basin (Australia) and determining its best-fit probability distribution model,” DOI:10.5675/ICWRER\_2013, pp. 43-52.