A Project Report on

Indoor Positioning System

Authors:
Ranganath Vaikuntham
Vamshi Bussa
Venkat Rohith T

Supervisors:
Dr. Raghu Kishore N

Submitted in partial fulfillment of the requirements
for the BTech degree.

MAHINDRA ECOLE CENTRALE
JNTU HYDERABAD
MAY 2018

CERTIFICATE

This is to certify that the project report entitled "Indoor Positioning System" submitted by
Ranganath Vaikuntham-14XJ1A0558
Vamshi Bussa-14XJ1A0511
Venkat Rohith T-14XJ1A0555
in partial fulfillment of the requirements for the BTech degree, embodies the work done by him under my supervision and guidance.

Dr. Raghu Kishore N
Assistant Professor
Mahindra Ecole Centrale
Hyderabad

ACKNOWLEDGEMENTS

We would like to thank Professor Raghu Kishore for his continuous guidance and support as a mentor providing valuable insight and direction throughout the course of the project. It was a wonderful experience working under him.

Team Indoor Positioning System.

ABSTRACT

Nowadays, developing indoor positioning systems (IPSs) has become an attractive research topic due to the increasing demands on location-based service (LBS) in indoor environments. Wi-Fi technology has been studied and explored to provide indoor positioning service for years in view of the wide deployment and availability of existing Wi-Fi infrastructures in indoor environments. Wi-Fi fingerprint-based localization has been one of the most attractive solutions, which is known to be free of extra infrastructure and specialized hardware. However, these IPSs suffer from major problem RSS fluctuations caused by unpredictable environmental dynamics. In this project, we intend to implement an Indoor Positioning System using various Artificial Intelligence and Machine learning algorithms, and to find the best among them.

The fast learning speed of Machine Learning/Training can reduce the time and manpower costs for the problem. Meanwhile, its self-learning ability enables the proposed localization algorithm to adapt in a timely manner to environmental dynamics. Experiments under specific environmental changes trainings are conducted to evaluate the performance. The simulation and experimental results would be able to show that the proposed localization algorithm can provide higher localizations accuracy than tradition.

CONTENTS

Introduction :

- ***What is Indoor Positioning System?***

  An indoor Positioning system (IPS) is a network of devices used to wirelessly locate objects or people inside a building. Without using satellites, an IPS depends on the nearby nodes (Access Points, Bluetooth devices, Visible light etc.), which are actively used to locate the target device. IPS can be localized to a Smartphone (or any other portable smart device). Like GPS, IPS can locate a person accurately.

- ***Characteristics of an ideal IPS:***

  Indoor localization systems using Wi-Fi infrastructure should ideally satisfy the following three requirements:

  - Deployable - They should be easily deployable on existing commodity Wi-Fi infrastructure without requiring any hardware or firmware changes at the access points (APs).
  - Universal – They should be able to localize any target device that has a commodity Wi-Fi chip and nothing else.
  - Accurate – They should be accurate, ideally as accurate as the best-known localization systems that use wireless signals.

- ***Why not GPS?***

  Satellite navigation is a system of satellites that provide autonomous "Geo-Spatial" positioning with global coverage may be termed a global navigation satellite system (GNSS or GPS). Challenges of GPS are – They don't work in the indoor environments and operate only in two dimensions. (Because of attenuation and scattering of micro signals by roof, walls etc.). Outdoors, navigation solely depends on GPS with accuracy 1 – 10 meters. GPS is vertically challenged and least accurate at pinpointing the elevation. Hence, Wi-Fi signals are used in IPS from an accuracy point of view.

- ***Location technology stack:***

A positioning technology stack is a layered structure of technical and environmental elements required to implement positioning systems and services. Signals, maps, positioning systems, and location-based applications are the major components constituting a positioning technology stack. The following is the pictorial representation of the stack along with the differences between the outdoor and indoor positioning environments from technological point of view.

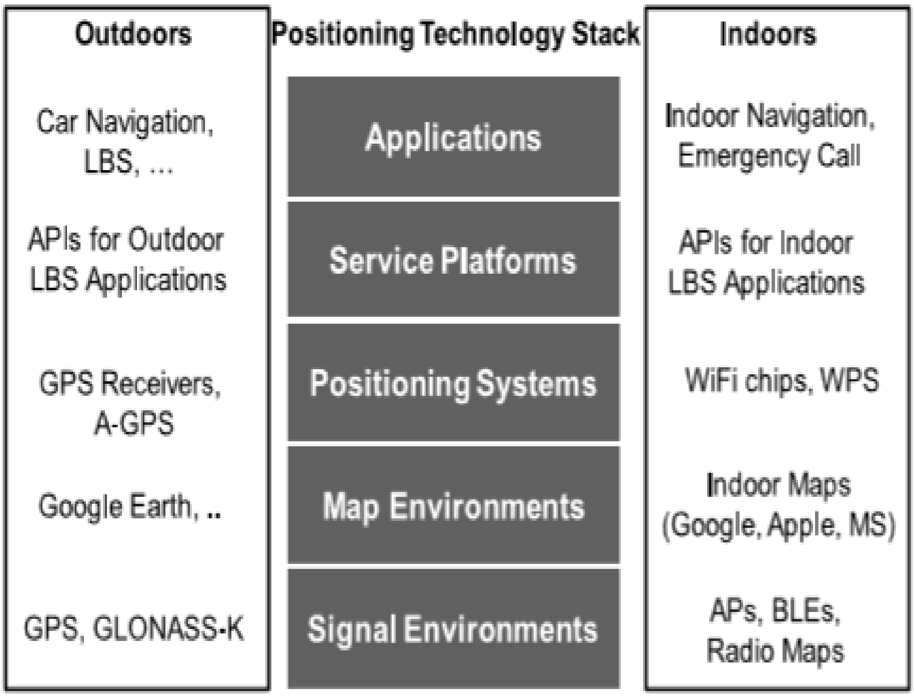


Figure 1: Location Technology Stack

- ***Steps to construct and IPS:***

  - Indoor maps should be prepared
  - A model of the area is required for radio map construction and more advanced techniques.
  - Radio maps are then constructed for the modelled indoor area by using one of the radio map construction methods.
  - Once the construction of a radio map is completed, an indoor positioning system is installed on top of the radio maps.

- Testing and evaluation must be performed to ensure the quality of the deployed indoor positioning systems.

- ***Uses:***

  - Shopping – Shop owners can activate an IPS which will send coupons and offers whenever the customer enters the mall.
  - Finding the mobile inside the house or building.
  - Company can track its employees inside the office thus monitoring if they leave premises.
  - Artefact information sent to user based on location inside a museum.
  - Philips works on the visible light indoor positioning system. It has an app which activates when a user is comes in the range on the visible light positioning device. If it is deployed in a shopping mall scenario, the customer is detected as he enters the mall and he gets the details of the mall like Map of the mall with item labelling in it, various offers on items etc. It can also be used to navigate to a particular place where the item is present or billing is automated depending on the items he selected.

Related work:

Traditional methods using triangle properties:

Based on the geometric properties of triangles, three methods can be used to calculate the position, namely received signal strength (RSS), angle of arrival (AOA) and time of arrival (TOA).

The basic principle of Triangulation is as follows. If the geographical coordinates (*xi, yi*) of three reference elements A, B, C are known, the absolute position E1 can be calculated by using either the length or the directions of *R*1, *R*2 and *R*3. Based on the information of the coverage area of an IPS, absolute, relative and proximity position information can be provided by the IPS using the triangulation method.

Each triangulation method has advantages and limitations. TOA is the most accurate technique, which can filter out multi-path effects in the indoor situations. However, it is complex to implement. RSS and TOA need to know the position of at least three references elements, such as A, B, C, to estimate the position of an object. AOA only requires two position measuring elements to perform location estimation. However, when the target object to be located is far away, the AOA method may contain some errors, which will result in lower accuracy.

- **RSSI based approaches**: This class of systems measures the RSSI from the target at multiple APs, combines them via triangulation along with a propagation model to locate the target. These systems are easy to deploy as RSSI values are easily available in current APs. Trilateration (sometimes called multi-lateration) techniques can be used to calculate the estimated client device position relative to the known position of access points.

  An example of this approach is the work done by Paramvir Bahl and Venkata Padmanabham of Microsoft Research at Colombia University [14]. They presented a technique 'RADAR' which is an RF based system. Basically, they use signal strength at multiple base stations to gauge the location which is an overlapping area of concern. Triangulation is done using both empirically determined and theoretically computed signal strength information.

  Though one of the cheapest and easiest methods to implement, its disadvantage is that it does not provide very good accuracy (median of 2-4m), because the RSSI measurements tend to fluctuate according to changes in the environment or multipath fading.

- **Time Based approaches:** Systems that use timestamps reported by Wi-Fi card scan obtain time of flight at a granularity of several nanoseconds, resulting in ranging error of few meters. In spite of hardware/firmware modifications to overcome coarse ToF estimates, the best-known systems achieve localization error of 2 m. Some systems applied super-resolution algorithms to obtain finer ToF estimates but require all the APs to be time synchronized which is hard to achieve using commodity Wi-Fi.

  Algorithms for joint estimation of AoA and ToF, to improve the accuracy of both the parameter estimates, have been developed, and tested in simulation. But these algorithms have been implemented in systems where the transmitter and receiver radios are time synchronized, which is not possible in commodity Wi-Fi deployments.

  In the ref [3], the Time of Flight value along with Angle of Arrival is taken into consideration. In this the intuition is that AoA and ToF estimates from the same path but different packets will be clustered together, but the diameter of each cluster (i.e., the tightness of each cluster) will be a function of the variations in AoA and ToF values for the corresponding path across packets.

- **Angle of Arrival based approaches:** With the proliferation of Wi-Fi APs with multiple antennas to support MIMO communications, antenna array-based techniques which use multiple antennas at the AP have gained interest recently. The basic idea of these systems is to calculate the AoAs of the multipath signals received at each AP, find the AoA of the direct path to the target, and then apply triangulation to localize.

  Typical computation of the AoA is done with the MUSIC algorithm. Assuming an antenna array of antennas equally spaced by a distance of and a signal arriving at the antenna array through propagation paths, an additional distance of is travelled by the signal to reach the second antenna of the array.

  AOA only requires two position measuring elements to perform location estimation. However, when the target object to be located is far away, the AOA method may contain some errors, which will result in lower accuracy. - [1]

  First, we would jointly estimate angle of arrivals(AoA) and time of flight (ToF), using small phase differences across all antennas and sub-carriers. The estimation step is to be followed by two additional steps, that include selecting the direct path from noisy estimates, and then combining noisy direct path estimates from multiple APs for triangulation to estimate the location - [SpotFi: Decimeter Level Localization Using Wi-Fi – Public Review Ashutosh Sabharwal Department of Electrical and Computer Engineering Rice University, Houston, TX ashu@rice.edu.]

Fingerprinting based approaches: Using pre-measured data, this technique is used to improve the accuracy of IPS measurements. Fingerprinting involves two phases:

offline training phase and online position determination phase. At first, data with respect to different places in the position estimation area is collected and measured for offline position estimation phase. Afterwards, during the online position determination phase, the location related data of target object is measured then compared with pre-measured data collected in the offline phase to find the case most similar in the database to make estimations. One key feature of this approach is generating fingerprinting maps using received signal strength from various AP's. Fingerprinting maps generated use the k-nearest neighbour algorithm to locate the target.

Fingerprinting positioning technique is proposed to improve the accuracy of indoor position measurements by using premeasured location related data. Fingerprinting includes two phases: offline training phase and online position determination phase. - [1].

An unsupervised learning-based fingerprint labeling technique was developed to construct radio maps by using crowdsourced fingerprints. It allows us to build the radio maps for most of the buildings in cities and villages at a very low cost. - [10]

Indoor navigation using Bluetooth: Since GPS does not work indoors, Bluetooth is a good alternative for indoor positioning and indoor navigation. Bluetooth beacons are able to send out signals, but they can't receive them. They are relatively cheap, can run on button cells up to two years and have a maximum range of 30 meters indoors. Accuracy is up to one meter. On the one hand they are used in client-based solutions, that is to say, positioning via app on the smartphone itself. In this case, Bluetooth must be activated on the device. On the other hand, server-based tracking solutions using beacons are possible as well.

In [15], Bluetooth Low Energy (BLE) technology which can be a very good alternative supplementing WiFi access points is used. Their combination will allow more accurate localization. The key advantage of BLE comprises low energy consumption which allows the transmitters—called beacons—to be powered continuously from batteries from months to years. This also makes it possible to place the beacons in the spots where WiFi access points would be difficult to power.

Visible light communication: VLC can be used as a positioning technology, mainly for inside areas. Special LED and fluorescent lamps send out indiscernibly flickering light which can be detected by a smartphone camera or a separate photo detector, which is for example attached to a shopping basket. This enables for example indoor navigation (via app) and tracking (analysis of motion profiles via app). In the future, VLC could also be used for wireless internet connection. Technically it works like that: Each lamp has its own ID which it compiles into pulsing light and sends to smartphones in the reception range. The app can access a map in which the lamps and their IDs are located. The incidence angle helps refine the position. Additional hardware such as beacons can fill in, where light doesn't advance.

In [11] the usage of Visible Light Technology like VLC using Light Emitting Diodes(LEDs) is explained. In this paper, an in-depth survey of the VLC-based-IPSs was made. VLC-based indoor positioning technologies such as LED technology, modulation method, and types of receivers are compared. A new taxonomy that two new classes, namely, sensor-assisted method and positioning optimization method are addressed.

PROBLEM STATEMENT:

To implement an IPS system using the Machine Learning techniques to find out the relative position of objects (sensors in this case) with respect to known fixed Access points(AP) using the techniques of Triangulation and Fingerprinting.

What are some of the questions we plan to answer?

● What is the percentage of Access Point information that should be known prior to mapping out the entire area?

● What level of accuracy and precision can be achieved?

● Which algorithm works best for our purpose?

We plan to compare the performance of various algorithms listed in the document like KNN, ANN, Autoencoder -Decoder etc. in terms of accuracy and precision.

As far as the percentage of Access point information required initially, we start out with minimum number of Access points and measure the performance. Then we increment the initial data available. Measure the performance again. Essentially, we reach a trade-off between optimal performance for least possible initial data.

Steps-Involved

There are two steps involved in implementation：

Data Set Collection:

There are various methods for collecting data (Fingerprints). A few of them are:

- ***Point-by-Point Manual Calibration (PMC***): In this method, an indoor area is partitioned into numerous pieces, i.e., locations, and then dedicated surveyors collect fingerprint samples point-by-point. But, PMC requires considerable time and effort.

- ***Walking Survey***: In the walking survey, the survey paths are planned in advance to guide the surveyors. The fingerprints are collected while the surveyors are walking along the paths carrying collection devices.

- ***Crowdsourcing approach***: In this method fingerprint samples are collected from numerous users without location information have been proposed to reduce the cost of constructing radio maps [14]. The crowdsourced fingerprint can be viewed as unlabeled data since the true locations at which he samples have been obtained are unknown.
- ***Semi-supervised Learning/ Pedestrians dead reckoning***: In this method, inertial sensors embedded in wireless devices, such as a three-axis accelerometer, a compass, and a gyroscope, can be used for estimating location of the users
- ***ULM (unsupervised learning-based location-labeling method)***: It is an extension to crowdsourcing method as it estimates the location labels of crowdsourced fingerprints collected from numerous smartphones. This method is distinguished from the semi-supervised learning- and the sensor-based methods because it does not require any explicit labeling effort or sensing data for reference.

The data set for our project may look like something mentioned in [16]. The fields of the data would be signal strengths form various Access points, device type, time stamp etc.

Classifying Query Fingerprint:

We plan to implement three different start-of-the-art approaches for comparison, which have been proposed to enhance the primary RSS fingerprinting.

1) Auto-Encoder Decoder: Reduced representation of sparse data containing multiple AP's could be essential in capturing vital information while reducing the complexity of computation.

2) KNN (K Nearest Neighbours): It forms fingerprints with temporally weighted RSS by applying an iterative recursive weighted average filter on training RSS samples.

3) Neural Networks: A middle ground between traditional AI approach like KNN and a pure ML method like GA, here after training we plan to use the ANN to classify novel entries of input parameters to predict the location.

Factors influencing result:

Impacts of AP number: We examine the impacts of AP number involved in the original RSS fingerprints. The reasons are that using too few AP's may result in loss of the spatial distinctiveness of fingerprints while using too many of them will as well involve in those low-quality AP's. Although we filter out -some apparently bad AP's by signal strengths, the selected AP's may not be all superior for location distinction. We need to remember that the minimum number of AP's to be used should be three for triangulation technique to work.

Impacts of neighbor number: Let us dig into the performance when using different numbers of neighbor fingerprints with r ranging from 2 to 5. More neighboring locations hold more stable spatial constraints, which would not be depicted well by too few neighboring fingerprints. However, when r further grows beyond a certain number, the performance would slightly degrade. This is because, as r increases, fingerprints from very distant locations will also be involved. These distant fingerprints would not contribute to, but in the contrast, will impair, the stableness of the generated FSG.

$$R_c = \frac{J_c - J_{c-1}}{J_{c+1} - J_c}$$

The number of clusters are chosen such that the ratio is maximized.

Implementation:

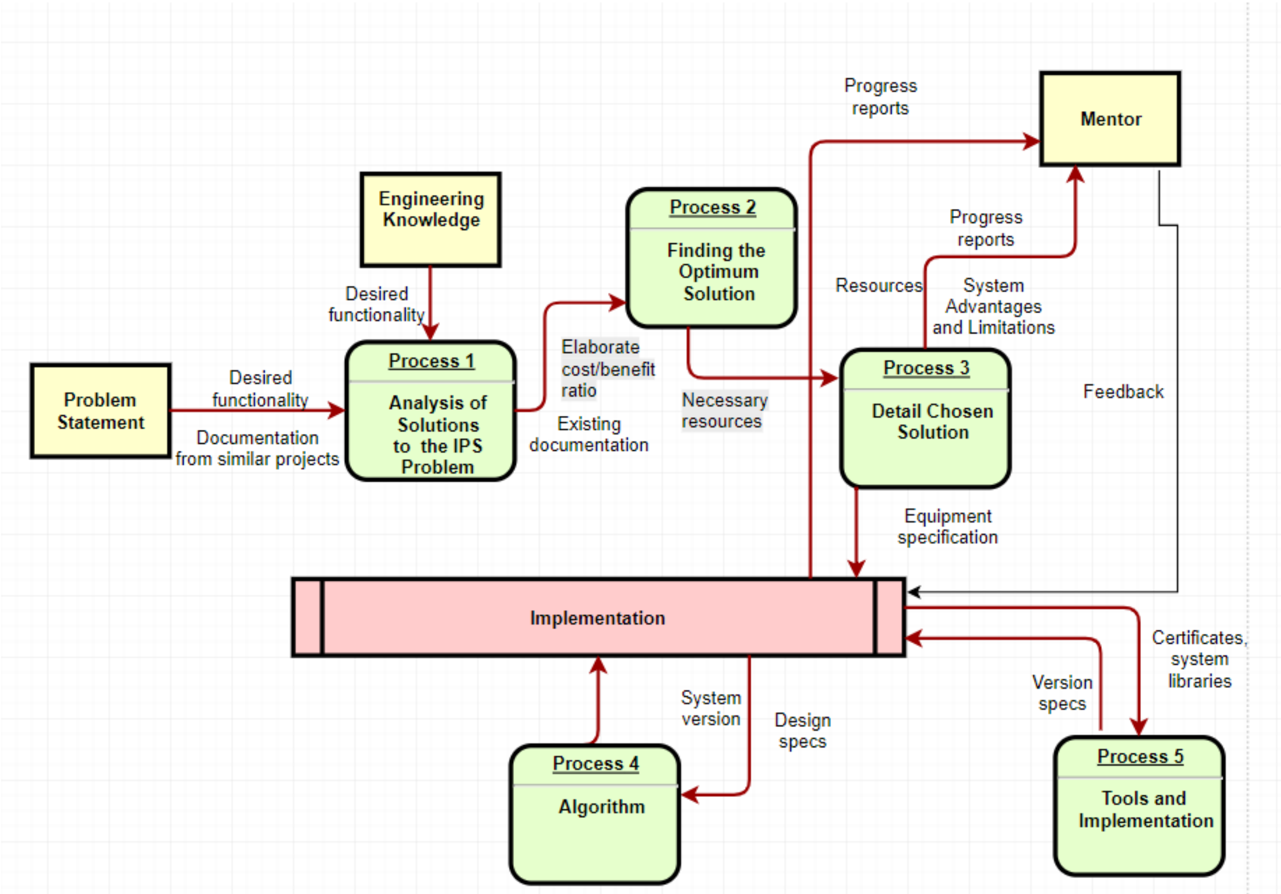
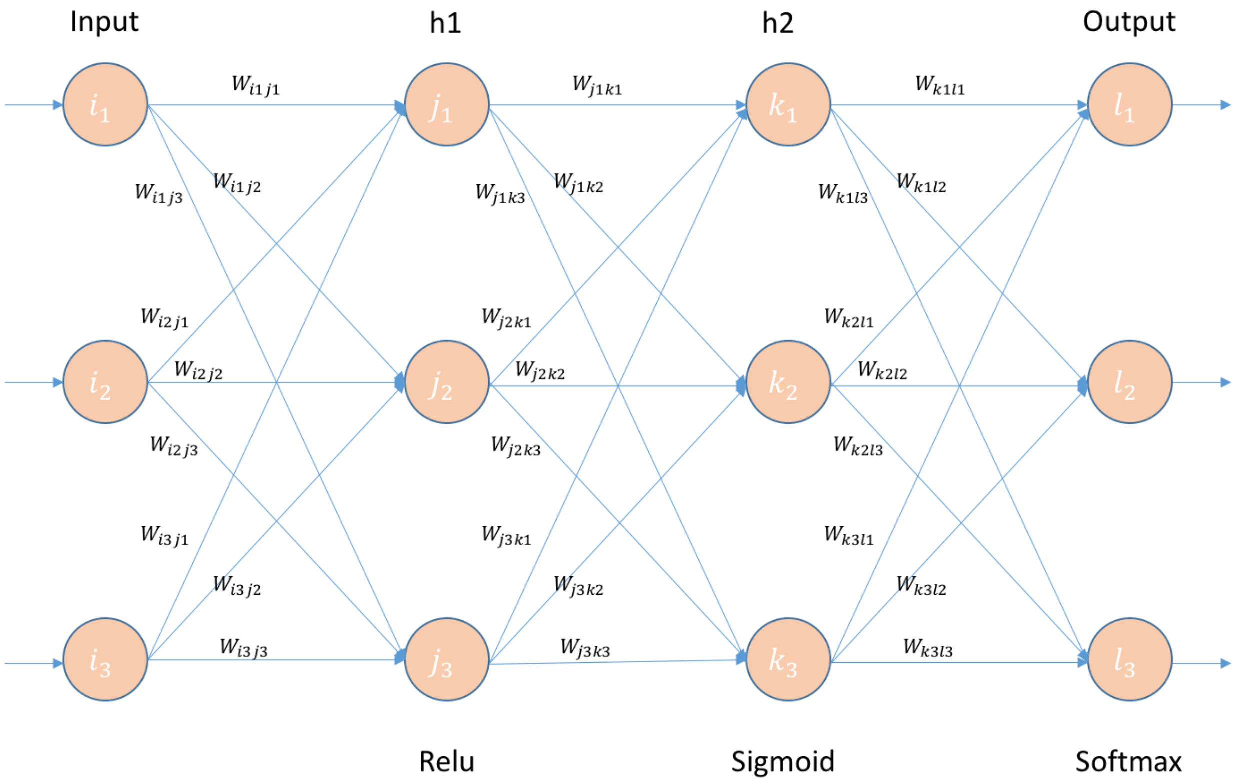


Figure: Flowchart of the process of implementation of our solution to the IPS problem

The following are implementations which were completed as parts to our solution:

Implementing Back Propagation in Artificial Neural Networks

The Back-propagation algorithm is a supervised learning method for multilayer feed-forward networks from the field of Artificial Neural Networks.



Feed-forward neural networks are inspired by the information processing of one or more neural cells, called a neuron. A neuron accepts input signals via its dendrites, which pass the electrical signal down to the cell body. The axon carries the signal out to synapses, which are the connections of a cell's axon to other cell's dendrites.

The principle of the backpropagation approach is to model a given function by modifying internal weightings of input signals to produce an expected output signal. The system is trained using a supervised learning method, where the error between the system's output and a known expected output is presented to the system and used to modify its internal state.

Technically, the backpropagation algorithm is a method for training the weights in a multilayer feed-forward neural network. As such, it requires a network structure to be defined of one or more layers where one layer is fully connected to the next layer. A standard network structure is one input layer, one hidden layer, and one output layer.

Back propagation can be used for both classification and regression problems. These steps will provide the foundation that you need to implement the back propagation algorithm from scratch and apply it to your own predictive modelling problems.

- Initialise Network
- Forward Propagate
- Back Propagate Error
- Train Network
- Predict

AUTOENCODER-DECODER

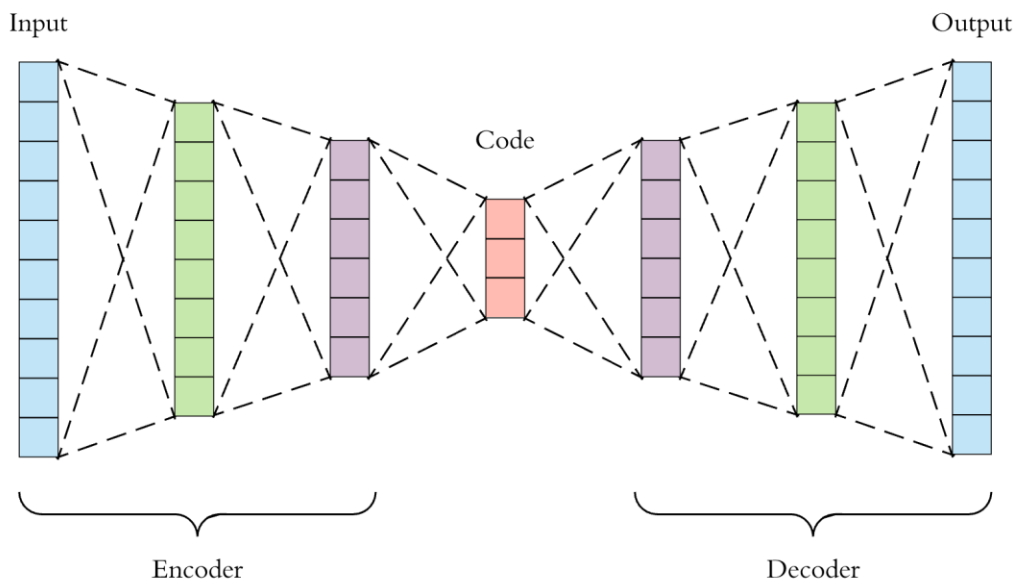
An **auto encoder** is an artificial neural network used for unsupervised learning of efficient codings. The aim of an auto encoder is to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction.

In these feedforward neural networks, the input is the same as the output. They compress the input into a lower-dimensional *code* and then reconstruct the output from this representation. The code is a compact "summary" or "compression" of the input, also called the *latent-space representation.*

An auto encoder consists of 3 components: encoder, code and decoder. The encoder compresses the input and produces the code, the decoder then reconstructs the input only using this code.

Architecture

Let's explore the details of the encoder, code and decoder. Both the encoder and decoder are fully-connected feedforward neural networks, essentially the ANNs. Code is a single layer of an ANN with the dimensionality of our choice. The number of nodes in the code layer (code size) is a *hyper parameter* that we set before training the auto encoder.



This is a more detailed visualisation of an auto encoder. First the input passes through the encoder, which is a fully-connected ANN, to produce the code. The decoder, which has the similar ANN structure, then produces the output only using the code. The goal is to get an output identical with the input. Note that the decoder architecture is the mirror image of the encoder. This is not a requirement but it's typically the case. The only requirement is the dimensionality of the input and output needs to be the same. Anything in the middle can be played with.

Auto encoders are trained the same way as ANNs via back propagation.

There are 4 hyper parameters that we need to set before training an auto encoder:

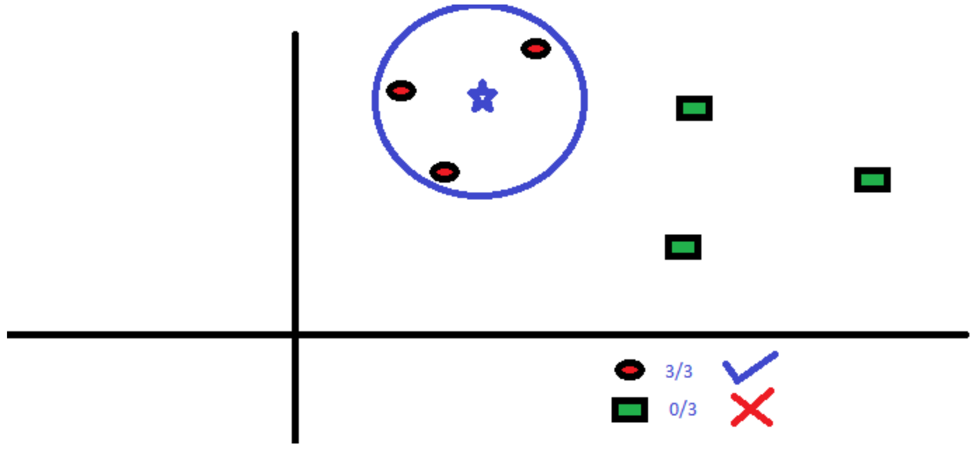
- Code size: is the number of nodes in the middle layer. The smaller the code size higher the compression, albeit increasing the possibility for more losses.
- Number of layers: we can choose the depth. The ideal depth would depend on the type of input and varies from problem to problem.
- Number of nodes per layer: it is a common practice that the number of nodes per layer decreases with each subsequent layer of the encoder and increases back in the decoder. Also, the decoder is symmetric to the encoder in terms of layer structure. Although it is ultimately our choice to decide these hyper parameters.
- Loss function: the error is measured through the loss function which is calculated by MSE (Mean Square Error). In case of inputs between [0,1] we use cross entropy.

KNN

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. It is commonly used for its easy of interpretation and low calculation time.

Let's take a simple case to understand this algorithm:

The task at hand is to find out the class of the blue star (BS) . BS can either be RC (Red Circle) or GS (Green Square) and nothing else. The "K" is KNN algorithm is the nearest neighbours we wish to take vote from. Let's say K = 3. Hence, we will now make a circle with BS as centre just as big as to enclose only three data points on the plane.



The three closest points to BS is all RC. Hence, with good confidence level we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbour went to RC. The choice of the parameter K is very crucial in this algorithm.

KNN for Regression:

The above algorithm explains the working of KNN for Classification problems. To apply the KNN technique for regression we simply find the 'K' nearest neighbours and then for every 'Label' to predict we simply average the label value of the neighbours.

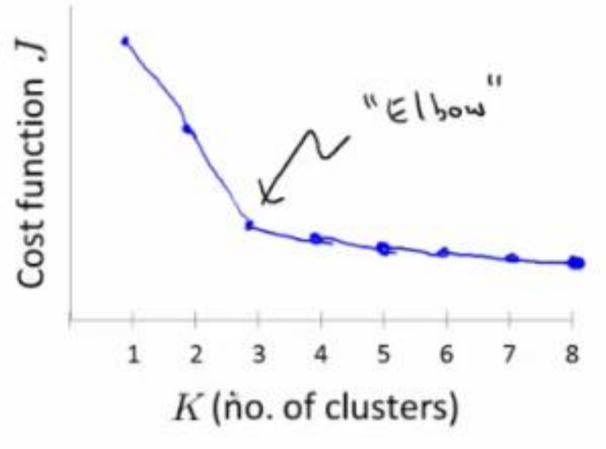
Corollary: - The Nearest Neighbour Algorithm

It is the KNN algorithm with the parameter 'K' equal to 1. Hence effectively only the nearest neighbour is being considered.

Choice of 'K' for KNN:

We select the optimum value of the neighbors 'K' by using the 'elbow method'. Here we check **Elbow Method**

- Vary K and compute cost function for the K.
- As K increases J minimum value should decrease (i.e. you decrease the granularity so centroids can better optimize)
- Plot this (K vs J())
- Look for the "elbow" on the graph
- Chose the "elbow" number of clusters



TENSORFLOW

TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google, often replacing its closed-source predecessor, DistBelief.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open source license on November 9, 2015.

TensorFlow is essentially a computational framework for building machine learning models. It provides a variety of different toolkits that allow you to construct models at your preferred level of abstraction. You can use lower-level APIs to build models by defining a series of mathematical operations. Alternatively, you can use higher-level APIs (like tf.estimator) to specify predefined architectures, such as linear regressors or neural networks.

The following figure shows the current hierarchy of TensorFlow toolkits:

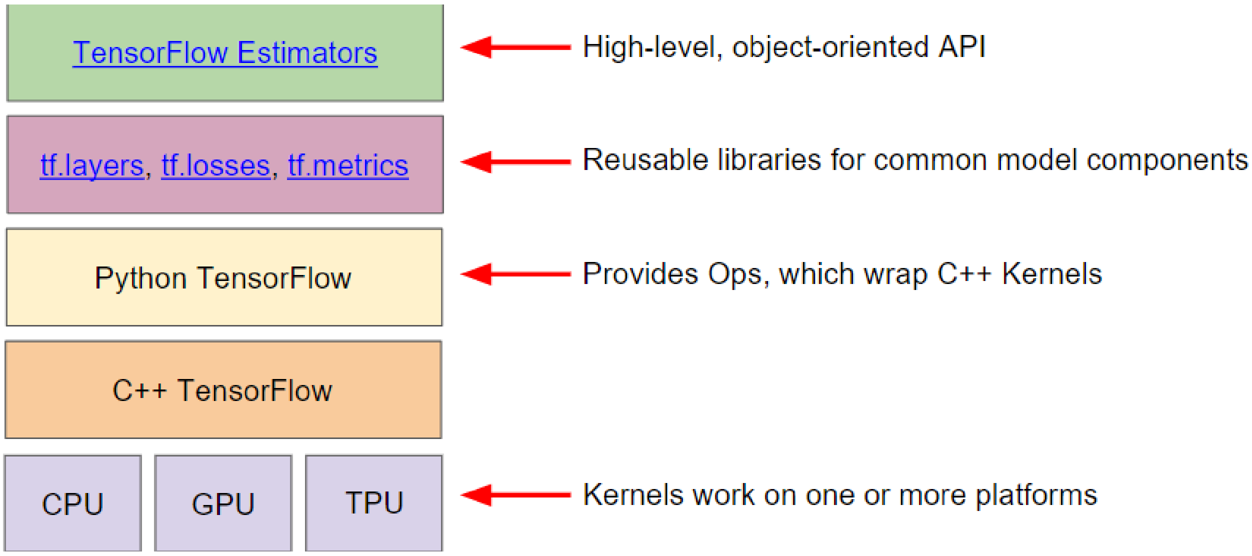


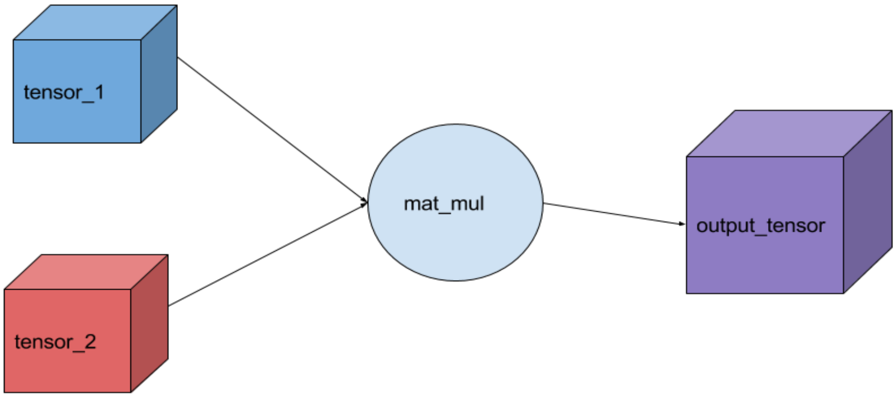
Figure 1. TensorFlow toolkit hierarchy.

Basic Computational Graph

Everything in TensorFlow is based on creating a computational graph. Think of a computational graph as a network of nodes, with each node known as an operation, running some function that can be as simple as addition or subtraction to as complex as some multi variate equation.

An Operation also referred to as op can return zero or more tensors which can be used later on in the graph. Here's a list of operations with their output for example

Each operation can be handed a constant, array, matrix or n-dimensional matrix. Another word for an n-dimensional matrix is a tensor, a 2-dimensional tensor is equivalent to a m x m matrix.



The above is creating two constant tensors and multiplying them together and outputting our result. This is a trivial example that demonstrates how you can create a graph and run the session. All inputs needed by the op are run automatically. They're typically ran in parallel. This session run actually causes the execution of three operations in the graph, creating the two constants then the matrix multiplication.

Adam: A Method for Stochastic Optimization

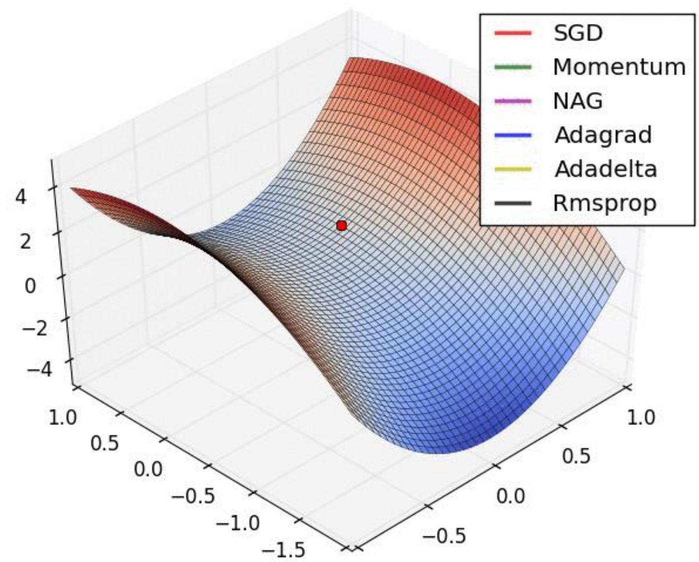
Adam optimizer, an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters. The method is also appropriate for non-stationary objectives and problems with very noisy and/or sparse gradients. The hyper-parameters have intuitive interpretations and typically require little tuning. It can be used to analyse the theoretical convergence properties of the algorithm and provide a regret bound on the convergence rate that is comparable to the best known results under the online convex optimization framework. Empirical results demonstrate that Adam works well in practice and compares favourably to other stochastic optimization methods.

How Does Adam Work?

Adam is different to classical stochastic gradient descent. Stochastic gradient descent maintains a single learning rate (termed alpha) for all weight updates and the learning rate does not change during training. A learning rate is maintained for each network weight (parameter) and separately adapted as learning unfolds. The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients.

Adam as combining the advantages of two other extensions of stochastic gradient descent. Specifically:

Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems).



Root Mean Square Propagation (RMSProp) that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy).

Adam realises the benefits of both AdaGrad and RMSProp.

Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance).

Specifically, the algorithm calculates an exponential moving average of the gradient and the squared gradient, and the parameters beta1 and beta2 control the decay rates of these moving averages.

The initial value of the moving averages and beta1 and beta2 values close to 1.0 (recommended) result in a bias of moment estimates towards zero. This bias is overcome by first calculating the biased estimates before then calculating bias-corrected estimates.

Adam Configuration Parameters :

Alpha: Also referred to as the learning rate or step size. The proportion that weights are updated (e.g. 0.001). Larger values (e.g. 0.3) results in faster initial learning before the rate is updated. Smaller values slow learning right down during training

Beta1: The exponential decay rate for the first moment estimates (e.g. 0.9).

Beta2: The exponential decay rate for the second-moment estimates (e.g. 0.999). This value should be set close to 1.0 on problems with a sparse gradient (e.g. NLP and computer vision problems).

Epsilon: Is a very small number to prevent any division by zero in the implementation (e.g. 10E-8).

Good default settings for the tested machine learning problems are alpha=0.001, beta1=0.9, beta2=0.999 and epsilon=10−8

The TensorFlow documentation suggests some tuning of epsilon:

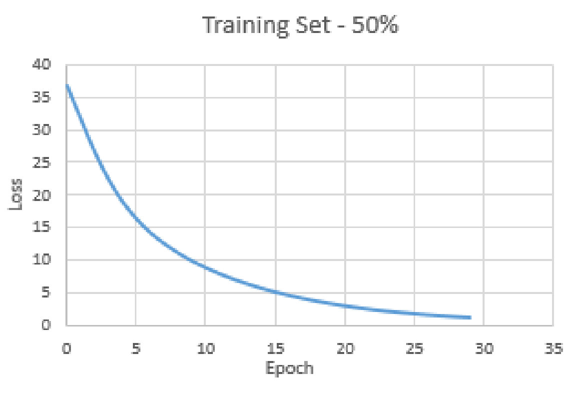
We can see that the popular deep learning libraries generally use the default parameters recommended by the paper.

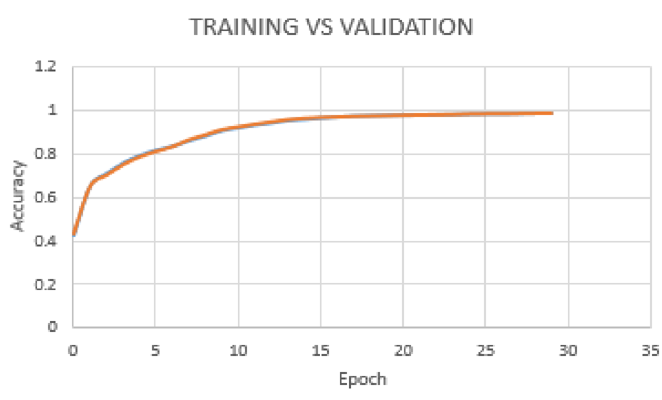
TensorFlow: learning_rate=0.001, beta1=0.9, beta2=0.999, epsilon=1e-08.

Results:

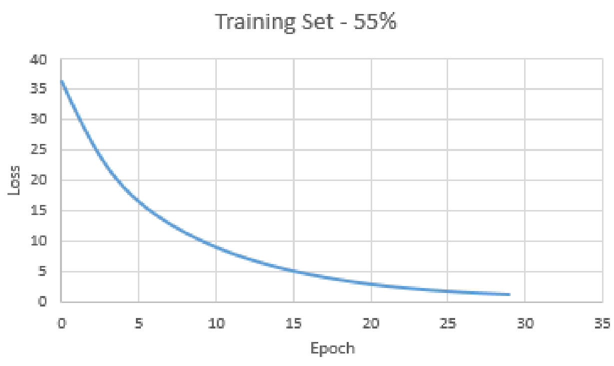
ANN- Results for Various ratios of Training and Validation

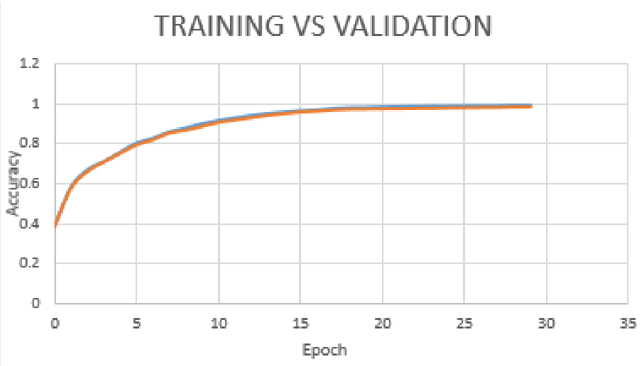
Training – 50%



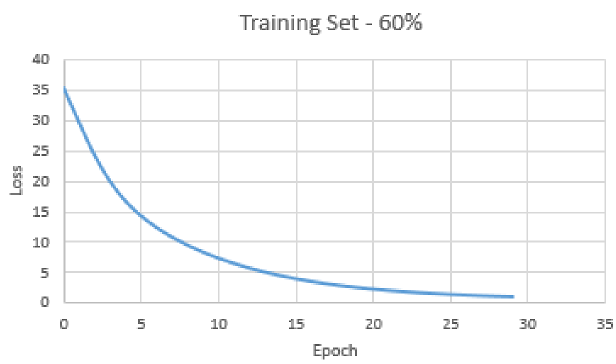


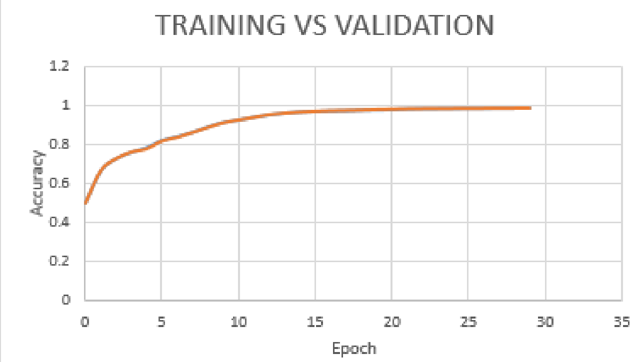
Training – 55%



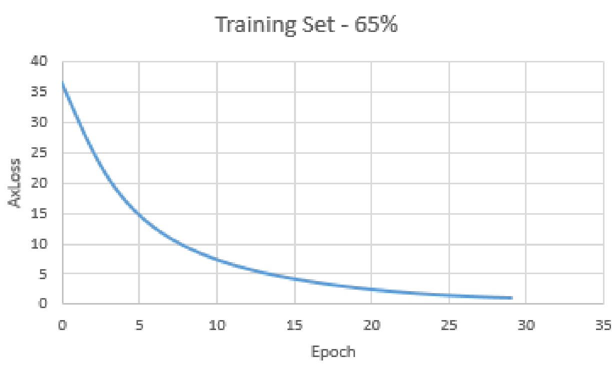


Training – 60%

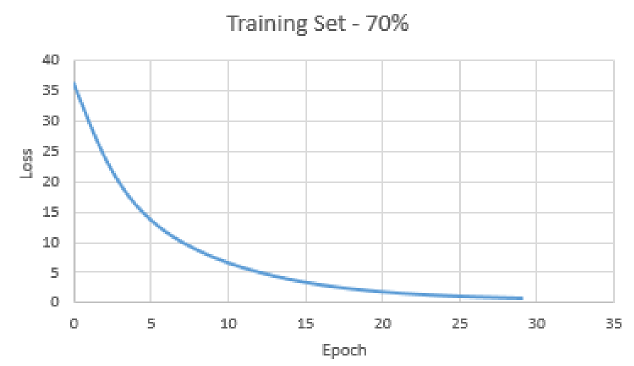




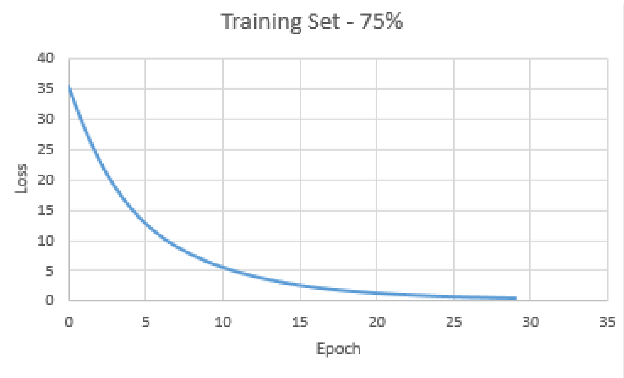
Training – 65%



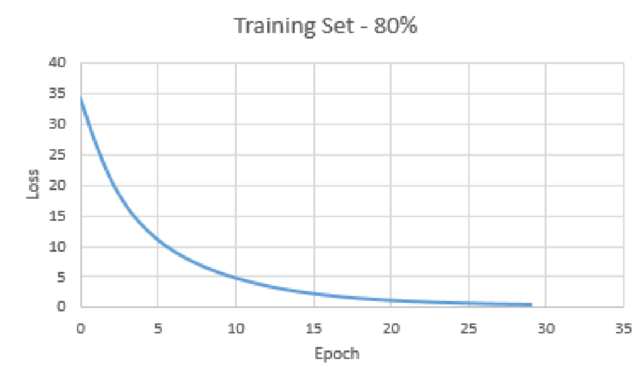
Training – 70%



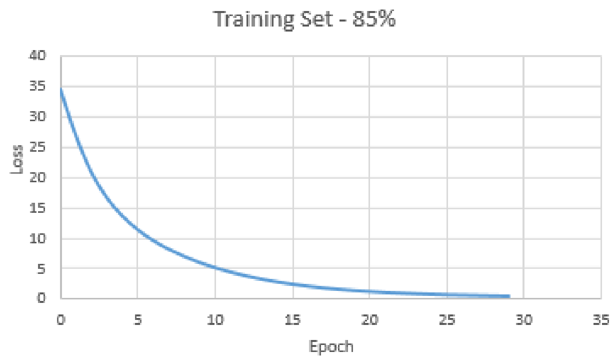
Training – 75%



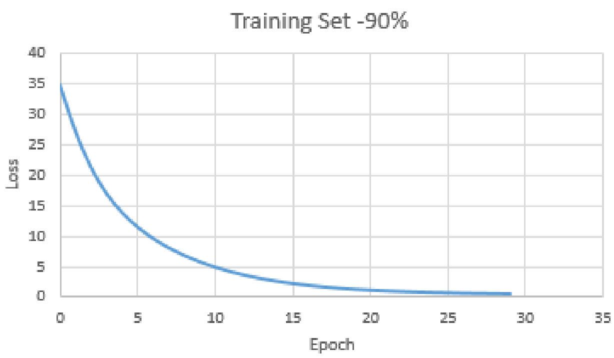
Training – 80%



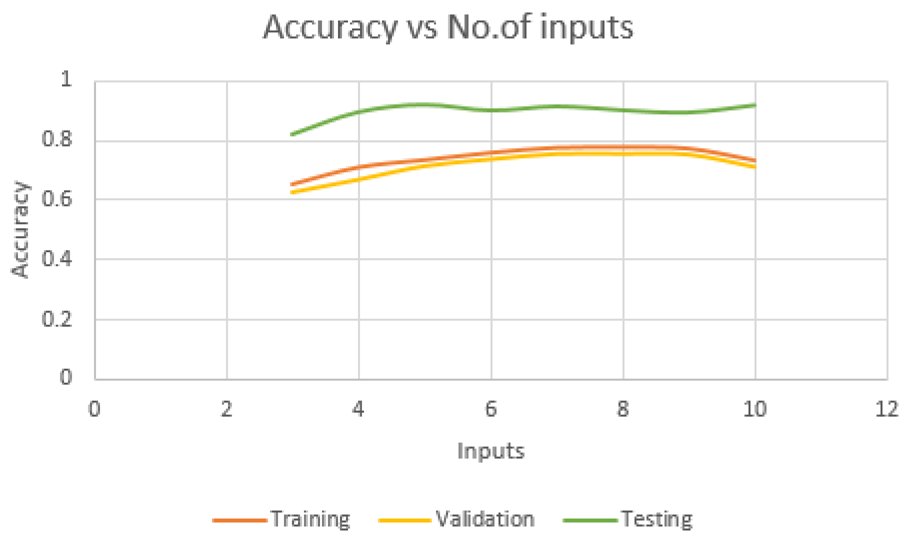
Training – 85%

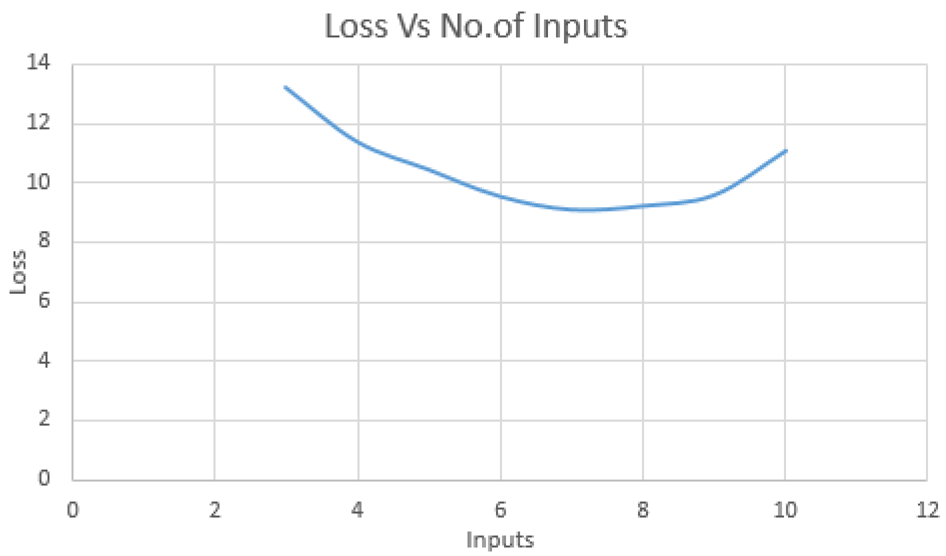


Training – 90%



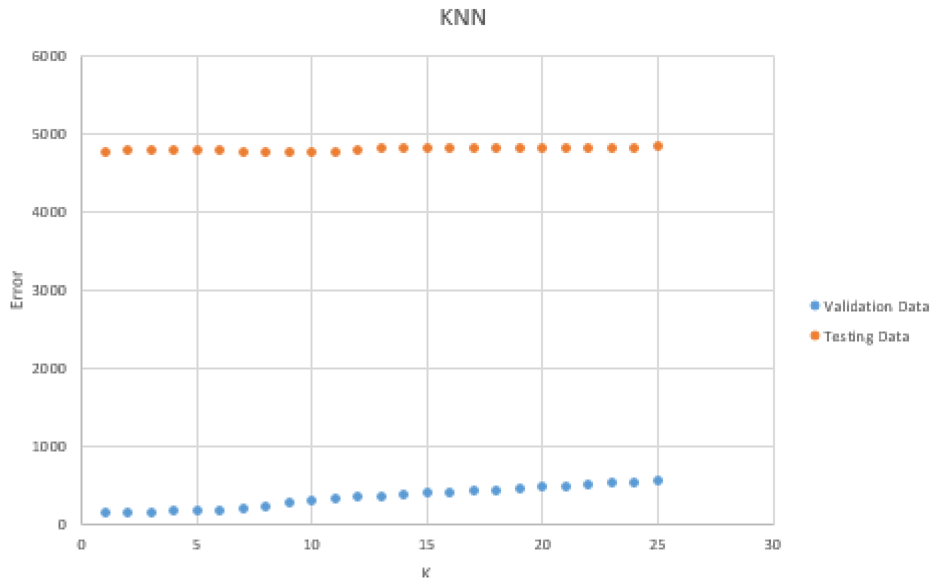
<u>Plain Dropouts</u>



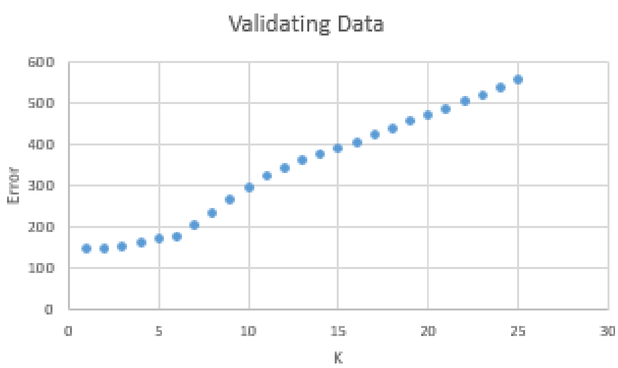


Loss Vs No.of Inputs
Loss
Inputs

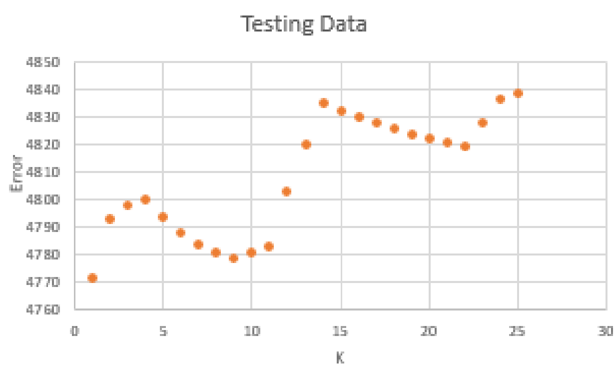
KNN



KNN
Error
K
Validation Data
Testing Data



Validating Data
Error
K

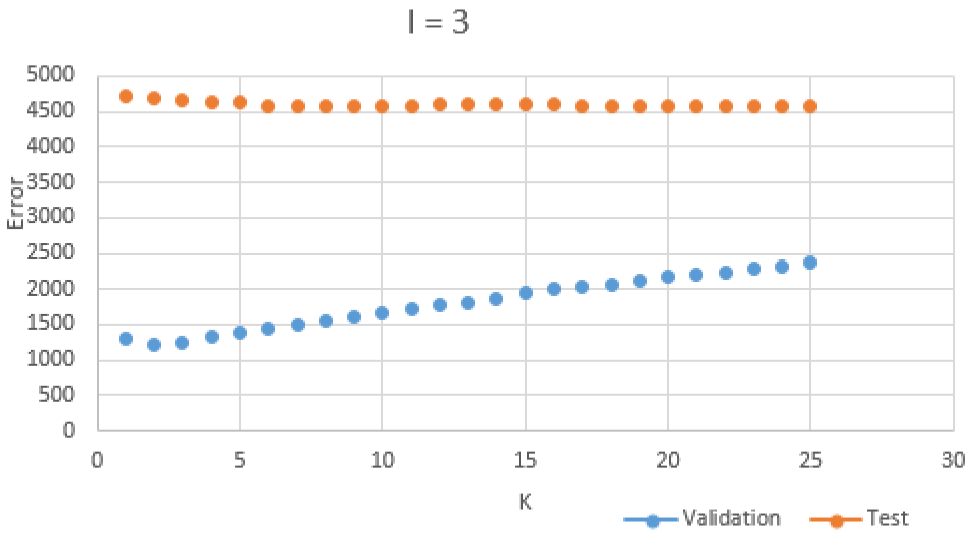


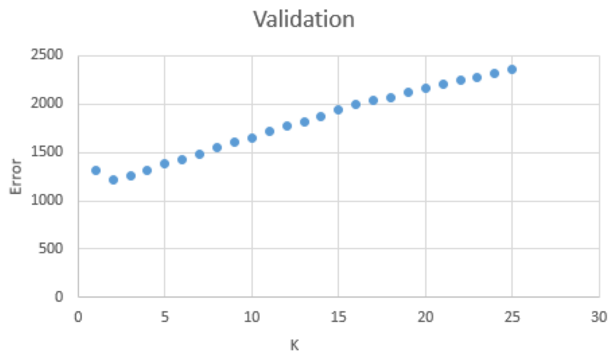
Testing Data
Error
K

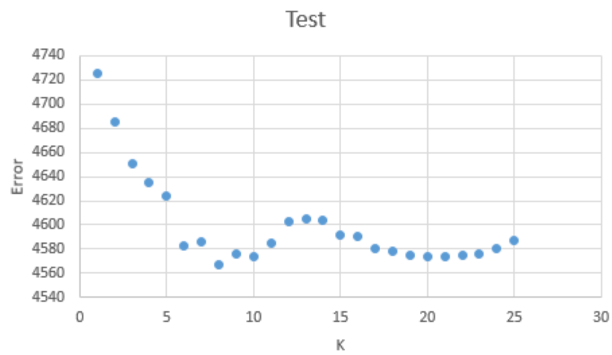
KNN with Dropouts:

I – No. of inputs

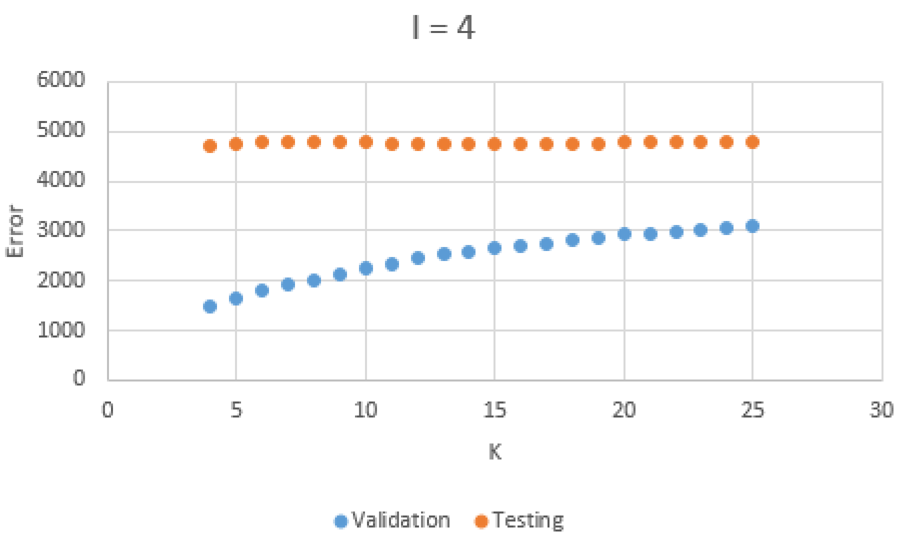
I = 3

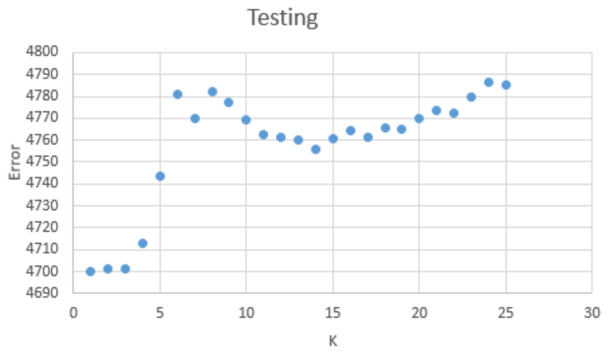


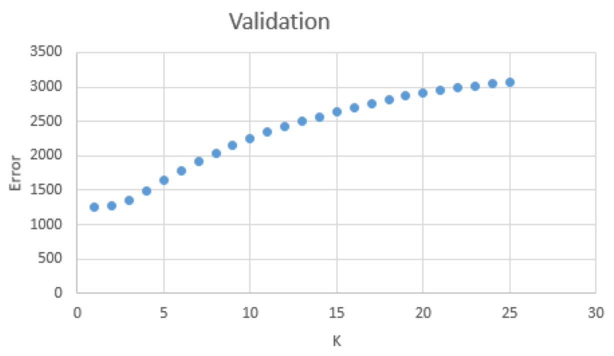




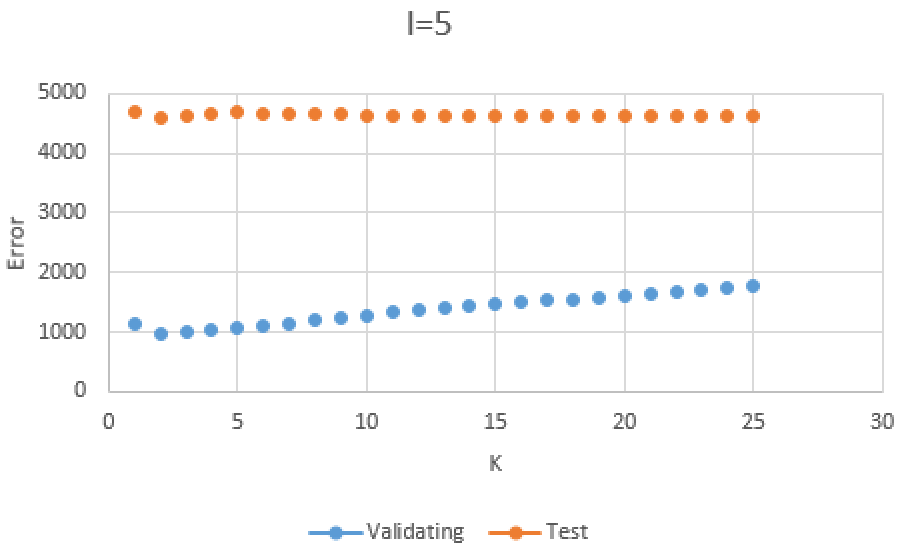
I = 4

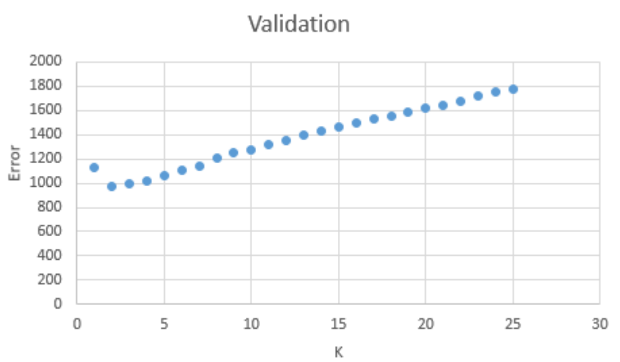


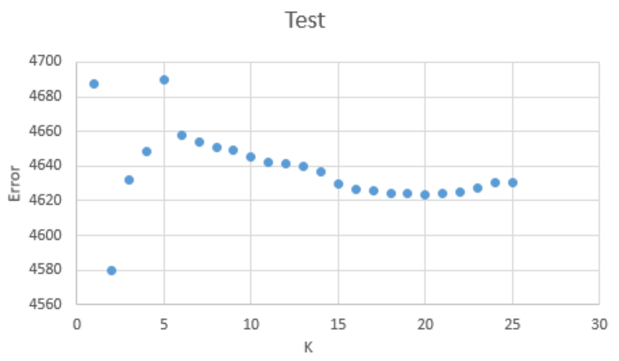

Testing
Error
K


Validation
Error
K

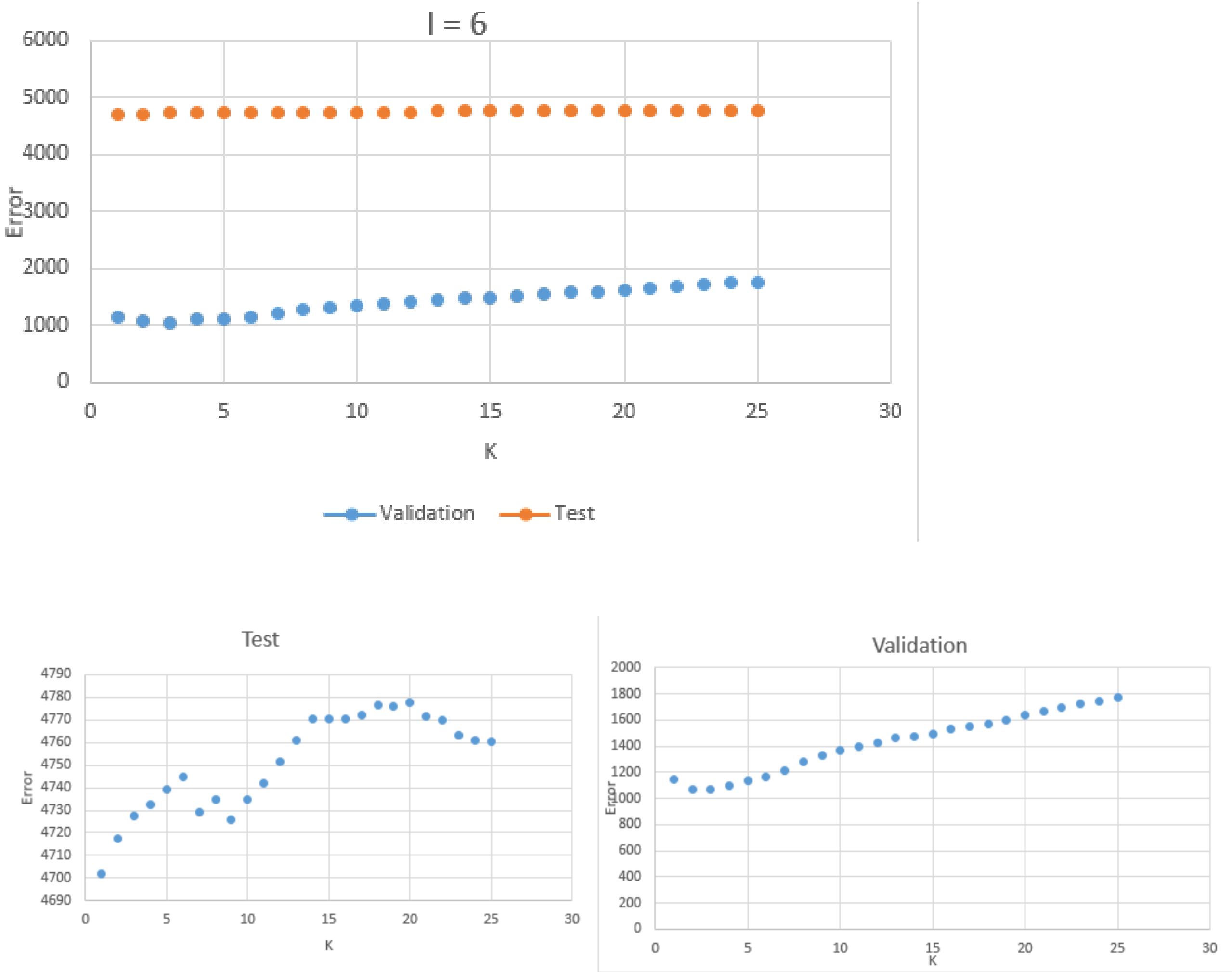
I = 5


I=5
Error
K
Validating
Test


Validation
Error
K

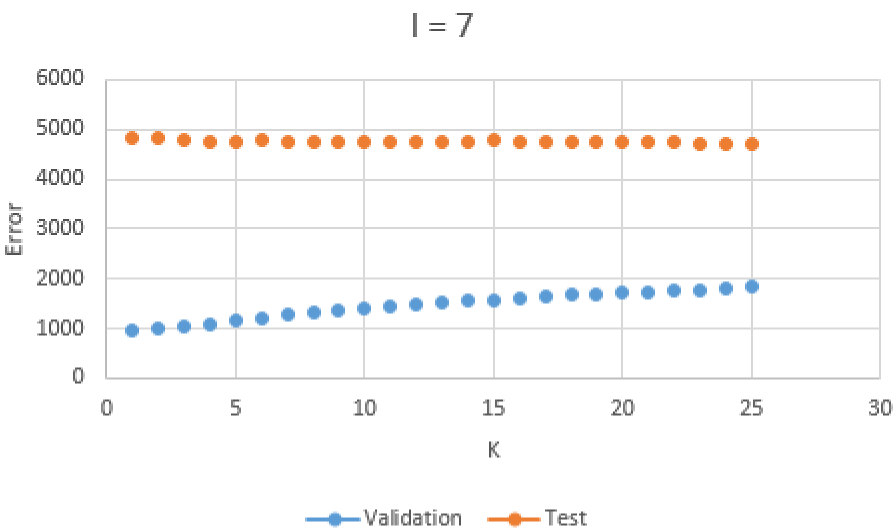

Test
Error
K

I = 6

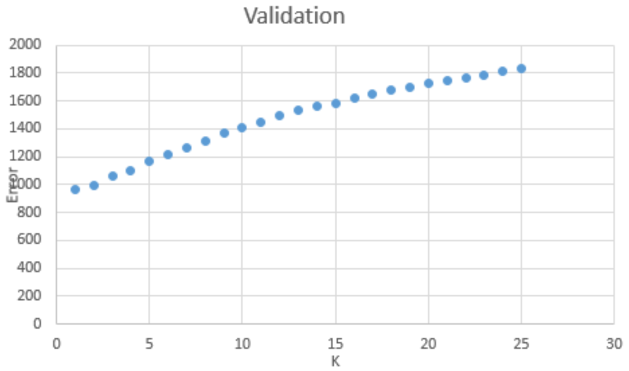


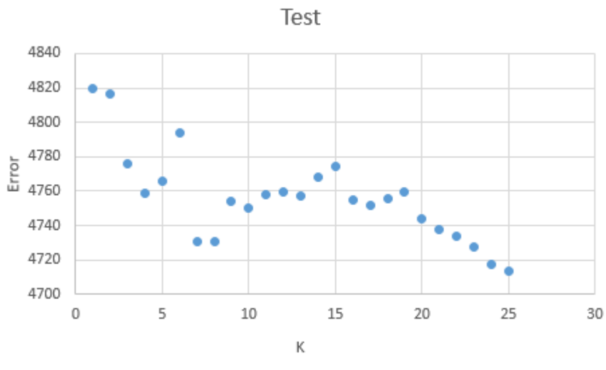
I = 6
6000
5000
4000
3000
2000
1000
0
Error
0
5
10
15
20
25
30
K
Validation
Test
Test
4790
4780
4770
4760
4750
4740
4730
4720
4710
4700
4690
Error
K
Validation
2000
1800
1600
1400
1200
1000
800
600
400
200
0
Error
K

I = 7

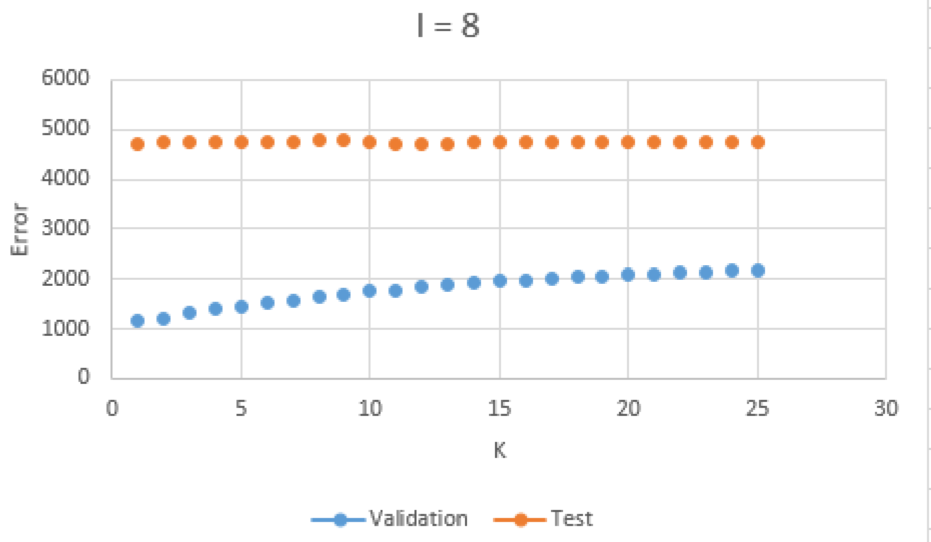


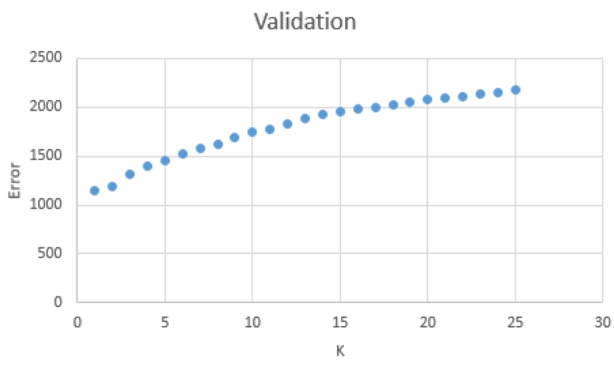
I = 7
6000
5000
4000
3000
2000
1000
0
Error
0
5
10
15
20
25
30
K
Validation
Test

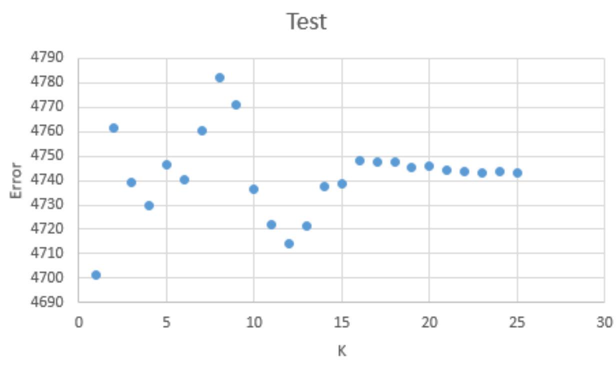

Validation
Error
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
5
10
15
20
25
30
K


Test
Error
4840
4820
4800
4780
4760
4740
4720
4700
0
5
10
15
20
25
30
K

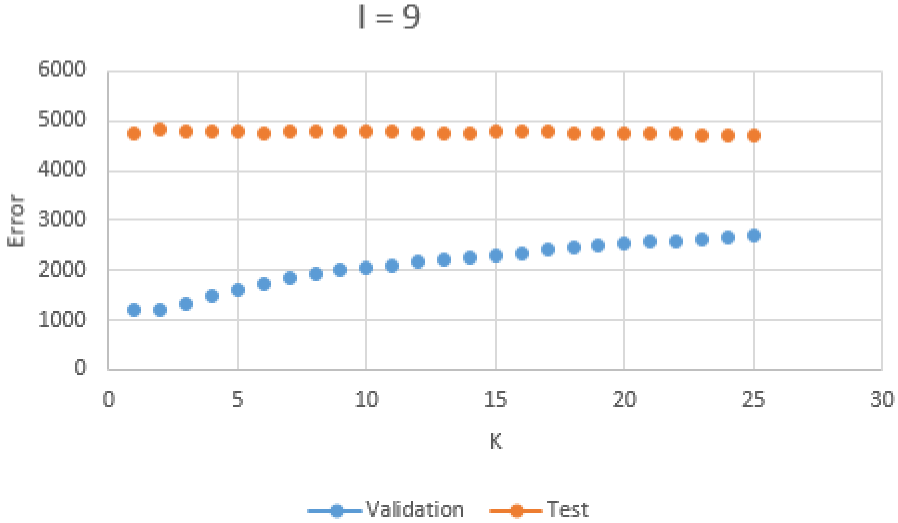
I = 8


I = 8
Error
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25
30
K
Validation
Test

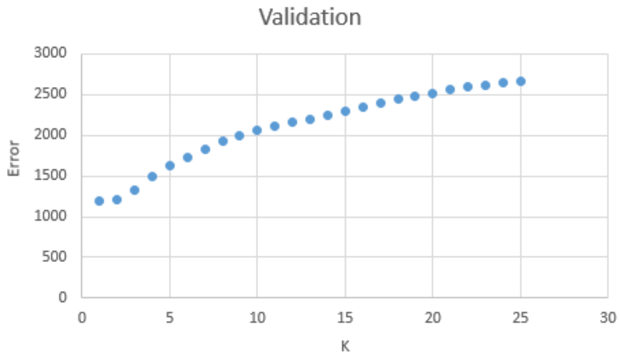

Validation
Error
2500
2000
1500
1000
500
0
0
5
10
15
20
25
30
K


Test
Error
4790
4780
4770
4760
4750
4740
4730
4720
4710
4700
4690
0
5
10
15
20
25
30
K

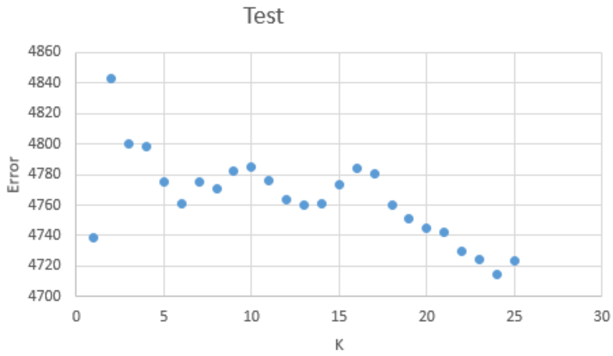
l = 9



l = 9
Error
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25
30
K
Validation
Test

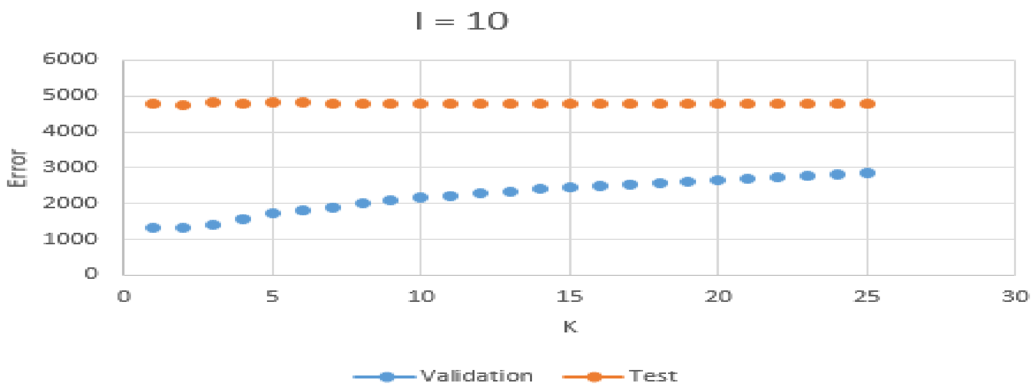


Validation
Error
3000
2500
2000
1500
1000
500
0
0
5
10
15
20
25
30
K

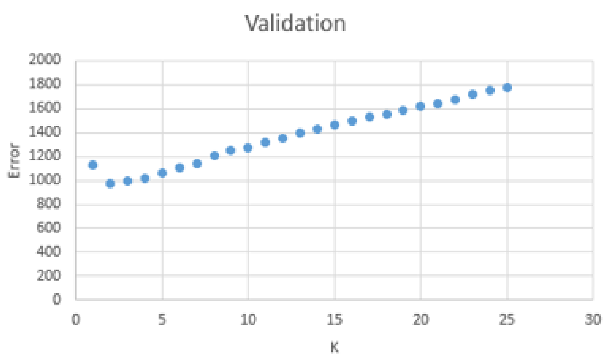


Test
Error
4860
4840
4820
4800
4780
4760
4740
4720
4700
0
5
10
15
20
25
30
K

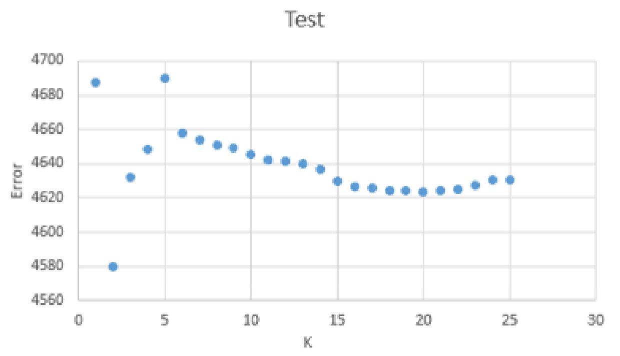
l = 10



l = 10
Error
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25
30
K
Validation
Test



Validation
Error
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
5
10
15
20
25
30
K



Test
Error
4700
4680
4660
4640
4620
4600
4580
4560
0
5
10
15
20
25
30
K

Conclusions/ Interpretations:

To solve our problem, we have arrived at the optimum algorithm to be of hierarchical approach. First, we predict the building and floor by using an Autoencoder – Decoder with a deep neural network. We achieved an accuracy of around 90%. All the correctly classified points were stored. Now, hierarchically regression was applied with the K-NN algorithm to predict the latitude and longitude of the point. Considerable accuracy was achieved in prediction as we can infer from the less magnitude of Mean Square Error.

We can see that optimum performance is achieved around the region of 80-20 % of Training and Validation error. How do we measure the performance? Looking at the 'Loss vs Epochs' graphs the one which has the highest gradient i.e. the one in which loss decays the fastest is deemed the best performing distribution. This is in sync with the traditional wisdom of selecting the training and validation data.

Now, we are in a position to answer the question of how many access points are required to locate a point efficiently with the least error. Based on the graphs which show the variation of accuracy and loss with respect to the number of inputs – Dropping out inputs as needed. We can see that the optimum performance – least error and maximum accuracy is achieved when 6-8 inputs are considered. Considering too many inputs would include far away access points thus diluting the predictive capability by including weak/erroneous signals. On the other hand including very few access points would exclude vital information which could aid in accurate prediction. Hence, our intuition is justified by the result.

References :

1). Y. Gu, A. Lo, I. Niemegeers, "A Survey of Indoor Positioning Systems for Wireless Personal Networks", IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 11, NO. 1, FIRST QUARTER 2009

2). Zou H, Lu X, Jiang H, Xie L. , A fast and precise indoor localization algorithm based on an online sequential extreme learning machine, NCBI PCM 2015

3). M .Kotaru, K.Joshi, D. Bharadia, S.Katti , "SpotFi: Decimeter Level Localization Using Wi-Fi" , Sigcomm Conference 2015

4). S.V. Ramani, Y. N. Tank, "Indoor Navigation on Google Maps and Indoor Localization Using RSS Fingerprinting ", International Journal of Engineering Trends and Technology (IJETT) – Volume 11 Number 4 - May 2014

5) Y. K. Cheng, H. J. Chou and R. Y. Chang, "Machine-Learning Indoor Localization with Access Point Selection and Signal Strength Reconstruction," *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, Nanjing, 2016, pp. 1-5.

6) W.W.Liang Li , R.A Ittis, M.Z.Win, "A smartphone localization algorithm using RSSI and inertial sensor measurement fusion", Global Communications Conference (GLOBECOM), 2013 IEEE

7) Dong Li, Baoxian Zhang, Zheng “A feature scaling-based k-nearest neighbor algorithm for indoor positioning system”, Global Communications Conference (GLOBECOM), 2014 IEEE

8) S. H. Jung, G. Lee and D. Han, "Methods and Tools to Construct a Global Indoor Positioning System," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1-14.

9) J. Luo, L. Fan and H. Li, "Indoor Positioning Systems Based on Visible Light Communication: State of the Art," in *IEEE Communications Surveys & Tutorials*, vol. PP, no. 99, pp. 1-1.

10) C. Wu, J. Xu, Z.Yang, N.D.Lane, Z,Yin “Gain Without Pain: Accurate WiFi-based Localization using Fingerprint Spatial Gradient”, Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2017

11) Data Set: https://archive.ics.uci.edu/ml/datasets/ujiindoorloc

12) P.Bahl and V.N Padmanabhan, "Radar: An In-Building RF-based User Location and Tracking System”, INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE

13) P Kriz, F Maly, and T Kozel, “Improving Indoor Localization Using Bluetooth Low Energy Beacons” , Hindawi Publishing Corporation Mobile Information Systems Volume 2016, Article ID 2083094

14) http://www.cedar.buffalo.edu/~srihari/CSE676/14%20Autoencoders.pdf

15)https://www.google.co.in/search?q=auto+encoder+decoder&oq=auto+encoder+decoder&aqs=chrome.0.0l6.7568j0j8&sourceid=chrome&ie=UTF-8

16) https://en.wikipedia.org/wiki/Autoencoder

17)https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

18)http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

19)https://www.google.co.in/search?q=knn+algorithm&oq=knn+algorithm&aqs=chrome..69i57j69i60j0l4.4037j0j8&sourceid=chrome&ie=UTF-8

20) http://www.saedsayad.com/k_nearest_neighbors.htm

21) https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

22)https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/

23) https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7

24)https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/

25) https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

26) https://medium.com/@nishantnikhil/adam-optimizer-notes-ddac4fd7218

27) http://neuralnetworksanddeeplearning.com/chap2.html

28) https://medium.com/@curiousily/tensorflow-for-hackers-part-iv-neural-network-from-scratch-1a4f504dfa8

29) https://www.datacamp.com/community/tutorials/tensorflow-tutorial

30) https://medium.com/@curiousily/tensorflow-for-hackers-part-i-basics-2c46bc99c930

31) https://medium.com/@camrongodbout/tensorflow-in-a-nutshell-part-one-basics-3f4403709c9d

32) https://ortus.rtu.lv/science/en/publications/12372/fulltext

33) https://pdfs.semanticscholar.org/182d/aec6d3781cbea47d06289889c3379e9ddc72.pdf