Satwik Ram Kodandaram Publication list and samples.

Google Scholar Profile: https://scholar.google.com/citations?user=pPFFz6AAAAAJ&hl=en

Reverse Recommendation: without user details

Publication Link: Reverse Recommendation System

Sign Language Recognition

Publication Link: Sign Language Recognition

Improving the Performance of Neural Networks

Publication Link: https://ijsret.com/wp-content/uploads/2021/09/IJSRET_V7_issue5_678.pdf

Masking private user information using Natural Language Processing

Publication Link: Masking private user information using Natural Language Processing

Smart Automation Attendance System using Neural Networks

Publication Link: https://scholar.google.com/scholar?oi=bibs&hl=en&cluster=14137946475275682476

Crop Infection Detection using YOLO

Publication Link: https://ijsret.com/wp-content/uploads/2021/09/IJSRET_V7_issue5_699.pdf

Machine Learning Algorithms

Paper Link: http://dx.doi.org/10.13140/RG.2.2.20266.54722

Sign Language Recognition

[1]Satwik Ram Kodandaram, [2]N Pavan Kumar [3] Sunil G L

[1]8th Sem UG Student,Visvesvaraya Technological University, Karnataka, India
[2]8th Sem UG Student,Visvesvaraya Technological University, Karnataka, India
[3]Assistant Professor,Visvesvaraya Technological University, Karnataka, India
E-mail: satwikram29@gmail.com, pavankumarpk031999@gmail.com, sunilgl.gls@gmail.com

ABSTRACT
Sign Language is mainly used by deaf (hard hearing) and dumbpeople to exchange information between their own community and with other people. It is a language where people use theirhand gestures to communicate as they can't speak or hear. Sign Language Recognition (SLR) deals with recognizing the hand gestures acquisition and continues till text or speech is generated for corresponding hand gestures. Here hand gestures for sign language can be classified as static and dynamic.However, static hand gesture recognition is simpler than dynamic hand gesture recognition, but both recognition is important to the human community.We can use Deep Learning Computer Vision to recognize the hand gestures by building Deep Neural Network architectures (Convolution Neural Network Architectures) where the model will learn to recognize the hand gestures images over an epoch.Once the model Successfully recognizes the gesture the corresponding English text is generated and then text can be converted to speech.This model will be more efficient and hence communicate for the deaf (hard hearing) and dump people will beeasier. In this paper, we will discuss how Sign Language Recognition is done using Deep Learning.

Index words*:*Hand Gestures; Sign Language Recognition; Convolution Neural Networks; Computer Vision; Text to Speech.

1. INTRODUCTION

Deaf (hard hearing) and dumb people use Sign Language (SL) [1] as their primary means toexpress their ideas and thoughtswith their own community and with other people with hand and body gestures.It has its own vocabulary, meaning, and syntax which is different from the spoken language or written language. Spoken language is a language produced by articulate sounds mapped against specific words and grammatical combinations to convey meaningful messages. Sign language uses visual hand and body gestures to convey meaningful messages.There are somewhere between 138 and 300 different types of Sign Language used around globally today. In India, there are only about 250 certified sign language interpreters for a deaf population of around 7 million. This would be a problem to teach sign language to the deaf and dumb people as there isa limitednumber of sign language interpreters exits today.Sign Language Recognition is an attempt to recognize these hand gestures and convert

Fig. 1. Sign Language Hand Gestures

them to the corresponding text or speech.Today Computer Vision and Deep Learning have gained a lot of popularity and many State of the Art (SOTA) models can be built. Using Deep Learning algorithms and Image Processing we can able to classify these hand gestures and able to produce corresponding text.An example of "A" alphabet in sign language notion to English "A" text or speech.

In Deep Learning Convolution Neural Networks (CNN) is the most popular neural network algorithm which is a widely used algorithm for Image/Video tasks. For Convolution Neural Networks (CNN) we have advanced architectures like LeNET-5 [2], and MobileNetV2 [3] where we can use these architectures to achieve the State of the Art (SOTA). We can use all these architectures and combine them using neural network ensemble techniques [4]. By this, we can achieve an almost 100% accurate model which will recognize the hand gestures. This model will be deployed in web frameworks like Django or a standalone application or embedded devices where the hand gestures arerecognized in the live camera and then converting them to text. This system will help deaf and dumb people to communicate easily.

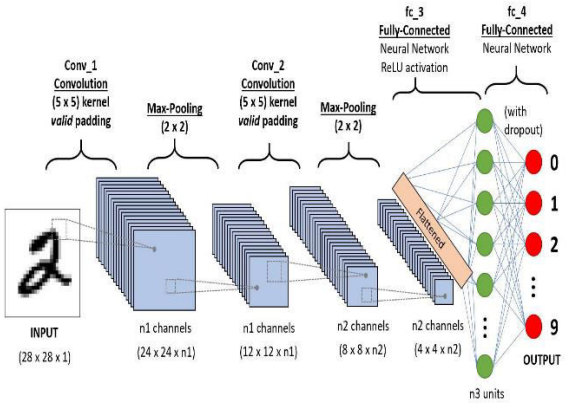


Fig. 2. Convolution Neural Networks

1.1 Motivation

The various advantages of building a Sign Language Recognition system includes:

- ✓ SignLanguage hand gestures totext/speech translation systems or dialog systems which are used in specific public domains such as airports, post offices, or hospitals.
- ✓ Sign Language Recognition (SLR) can help to translate the video to text or speech enables inter-communication between normal and deaf people.

1.2 Problem Statement

Sign language uses lots of gestures so that it looks like movement language which consists of a series of hands and arms motions. For different countries, there are different sign languages and hand gestures.Also, itis noted that some unknown words are translated by simply showing gestures for each alphabet in the word. In addition, sign language also includes specific gestures to each alphabet in the English dictionary and for each number between 0 and 9.

Based on these sign languages are made up of two groups, namely static gesture, and dynamic gesture. The static gesture is used for alphabet and number representation, whereas the dynamic gesture is used for specific concepts. Dynamic also includes words, sentences, etc. The static gesture consists of hand gestures, whereas the latter includes motion of hands, head, or both. Sign language is a visual language and consists of 3 major components, such as finger-spelling, word-level sign vocabulary, and non-manual features. Finger-spelling is used to spell words letter by letter and convey the message whereas the latter is keyword-based. But the design of a sign language translator is quite challenging despite many research efforts during the last few decades. Also, even the same signs have significantly different appearances for different signers and different viewpoints. This work focuses on the creation of a static sign language translator by using a Convolutional Neural Network. We created a lightweight network that can be used with embedded devices/standalone applications/web applications having fewer resources.

1.3 Objectives

The main objectives of this project are to contribute to the field of automatic sign language recognition and translation to text or speech. In our project, we focus on static sign language hand gestures. This work focused on recognizing the hand gestures which includes 26 English alphabets (A-Z) and 10 digits (0-9) using Deep Neural Networks (DNN). We created a convolution neural networks classifier that can classify the hand gestures into English alphabets and digits. We have trained the neural network under different configurations and architectures like LeNet-5 [2], MobileNetV2 [3], and our own architecture. We used the horizontal voting ensemble technique to achieve the maximum accuracy of the model. We have also created a web application using Django Rest Frameworks to test our results from a live camera.

2. LITERATURE REVIEW

2.1Real-time sign language fingerspelling recognition using convolutional neural networks from depth map [5].

This works focuses on static fingerspelling in American Sign Language A method for implementing a sign language to text/voice conversion system without using handheld gloves and sensors, by capturing the gesture continuously and converting them to voice.

In this method, only a few images were captured for recognition. The design of a communication aid for the physically challenged.

2.2Design of a communication aid for physically challenged [6].

The system was developed under the MATLAB environment. It consists of mainly two phases via training phase and the testing phase. In the training phase, the author used feed-forward neural networks. The problem here is MATLAB is not that efficient and also integrating the concurrent attributes as a whole is difficult.

2.3 American Sign Language Interpreter System for Deaf and Dumb Individuals [7].

The discussed procedures could recognize 20 out of 24 static ASL alphabets. The alphabets A, M, N, and S couldn't be recognized due to the occlusion problem. They have used only a limited number of images.

3. IMPLEMENTATION

3.1 Dataset

We have used multiple datasets and trained multiple models to achieve good accuracy.

3.1.1 ASL Alphabet

The data is a collection of images of the alphabet from the American Sign Language, separated into 29 folders that represent the various classes.

The training dataset consists of 87000 images which are 200x200 pixels. There are 29 classes of which 26 are English alphabets A-Z and the rest 3 classes are SPACE, DELETE, and, NOTHING. These 3 classes are very important and helpful in real-time applications.

3.1.2Sign Language Gesture Images Dataset

The dataset consists of 37 different hand sign gestures which include A-Z alphabet gestures, 0-9 number gestures, and also a gesture for space which means how the deaf (hard hearing) and dumb people represent space between two letters or two words while communicating.

Each gesture has 1500 images which are 50x50 pixels, so altogether there are 37 gestures which means there 55,500 images for all gestures. Convolutional Neural Network (CNN) is well suited for this dataset for model training purposes and gesture prediction.

3.2 Data Pre-processing

An image is nothing more than a 2-dimensional array of numbers or pixels which are ranging from 0 to 255.Typically, 0 means black, and 255 means white. Image is defined by mathematical function f(x,y) where 'x' represents horizontal and 'y' represents vertical in a coordinate plane. The value of f(x, y) at any point is giving the pixel value at that point of an image.

Image Pre-processing is the use of algorithms to perform operations on images. It is important to Pre-process the images before sending the images for model training. For example, all the images should have the same size of 200x200 pixels. If not,the model cannot be trained.

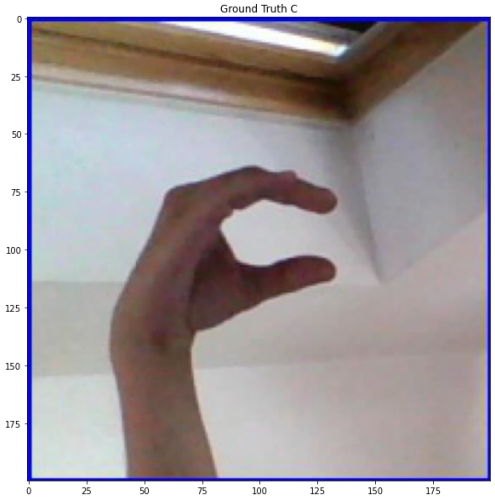


Fig. 3. Sample Image without Pre-processing

Fig. 4. Pre-Processed Image

The steps we have taken for image Pre-processing are:

- ✓ Read Images.
- ✓ Resize or reshape all the images to the same
- ✓ Remove noise.
- ✓ All the image pixels arrays are converted to 0 to 255 by dividing the image array by 255.

3.3 Convolution Neural Networks (CNN)

Computer Vision is a field of Artificial Intelligence that focuses on problems related to images andvideos. CNN combined with Computer vision is capable of performing complex problems.

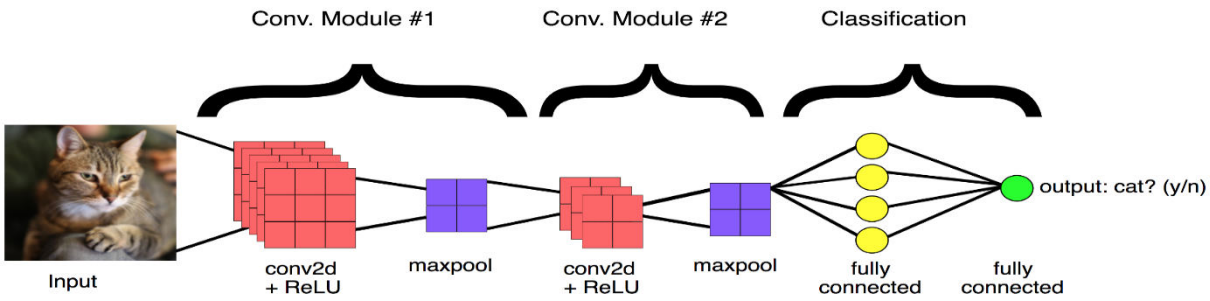


Fig. 5. Working of CNN

The Convolution Neural Networks has two main phases namely feature extraction and classification.

A series of convolution and pooling operations are performed to extract the features of the image.

The size of the output matrixdecreases as we keep on applying the filters.

Size of new matrix = (Size of old matrix — filter size) +1

A fully connected layer in the convolution neural networks will serve as a classifier.

In the last layer, the probability of the class will be predicted.

The main steps involved in convolution neural networks are:

1. Convolution
2. Pooling
3. Flatten
4. Full connection

3.3.1 Convolution

Convolution is nothing but a filter applied to an image to extract the features from it. We will use different filters to extract features like edges, highlighted patterns in an image. The filters will be randomly generated.

What this convolution does is, creates a filter of some size says 3x3 which is the default size. After creating the filter, it starts performing the element-wise multiplication starting from the top left corner of the image to the bottom right of the image.

The obtained results will be extracted feature.

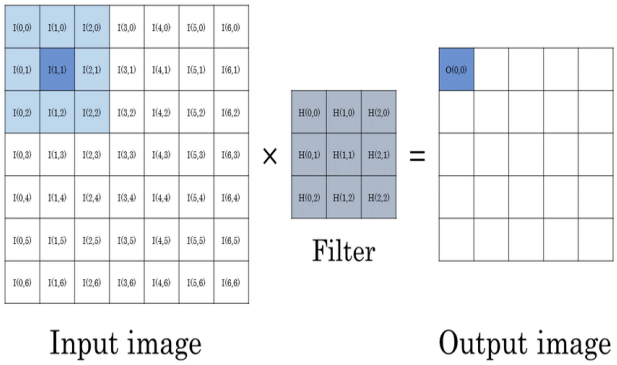


Fig. 6. Convolution

Fig. 6. Feature Extraction

3.3.2 Pooling

After the convolution operation, the pooling layer will be applied. The pooling layer is used to reduce the size of the image. There are two types of pooling:

1. Max Pooling
2. Average Pooling

3.3.2.1 Max pooling

Max pooling is nothing but selecting the maximum pixel value from the matrix.

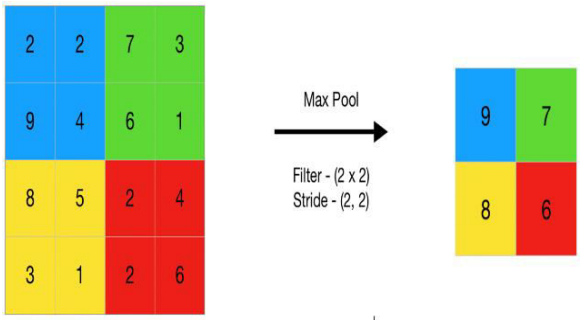


Fig. 7. Max Pooling

This method is helpful to extract the features with high importance or which are highlighted in the image.

3.3.2.2 Average pooling

Unlike Max pooling, the average pooling will take average values of the pixel

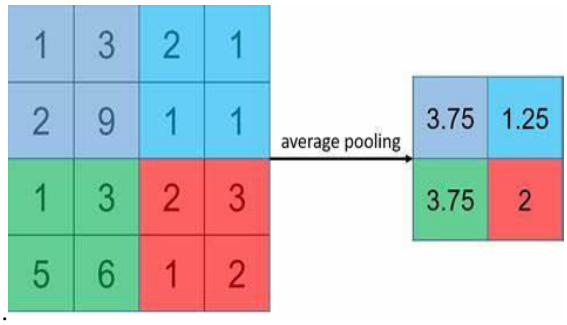


Fig. 8. Average Pooling

In most cases, max pooling is used because its performance is much better than average pooling.

3.3.3 Flatten

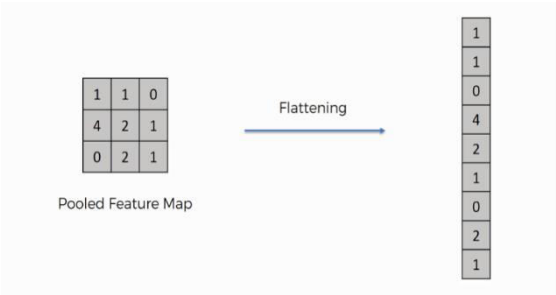
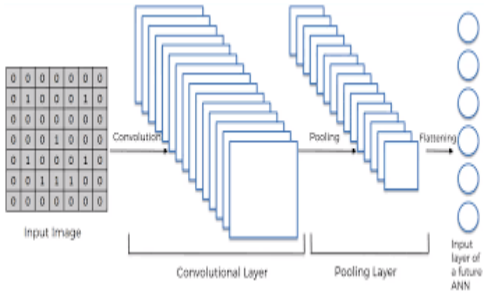


Fig. 9. Flatten

The obtained resultant matrix will be in muti-dimension. Flattening is converting the data into a 1-dimensional array for inputting the layer to the next layer. We flatten the convolution layers to create a single feature vector.



3.3.4 Full Connection

Fig. 10. Full Connection

A fully connected layer is simply a feed-forward neural network. All the operations will be performed and prediction is obtained. Based on the ground truth the loss will be calculated and weights are updated using gradient descent backpropagation algorithm.

3.4 Convolution Neural Network (CNN) Architectures

3.4.1 LeNet-5

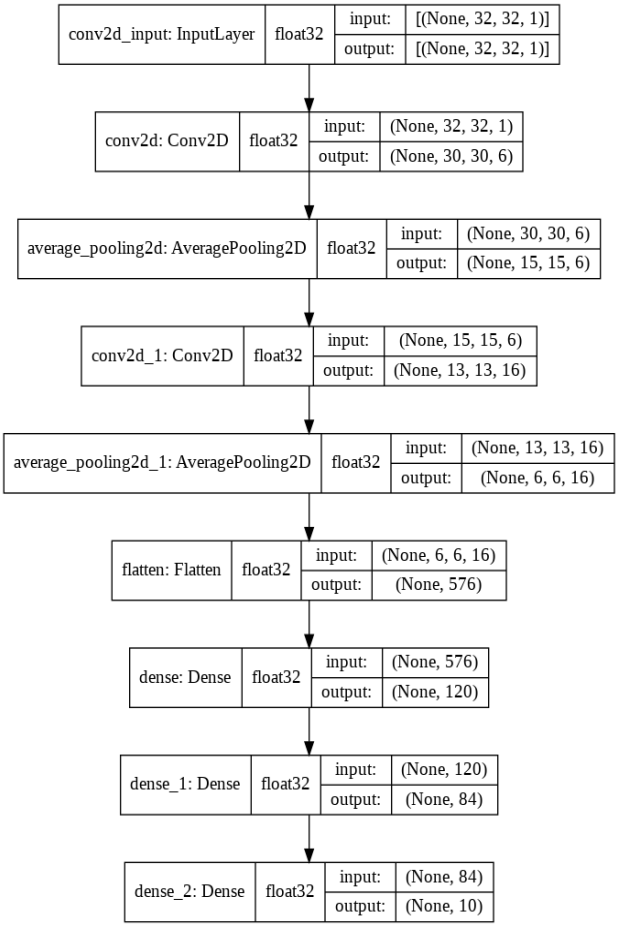


Fig. 11. LeNet-5 Implementation

The LeNet-5 [2] architecture consists of two pairs of convolutional and average pooling layers, followed by a flattening convolutional layer, then two fullyconnected layers, and finally a SoftMax classifier.

3.4.2 MobileNetV2

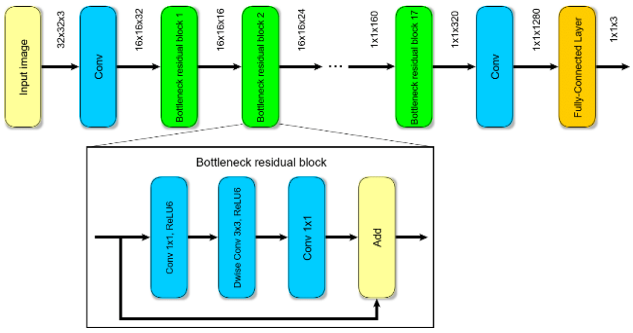


Fig. 12. MobileNetV2 Implementation

MobileNetV2 [3] is a convolutional neural network architecture that performs well on mobile devices.The architecture of MobileNetV2 contains thefully convolution layer with 32 filters, followed by 19 residual bottleneck layers. This network is lightweight and efficient.

3.4.3 Own Architecture

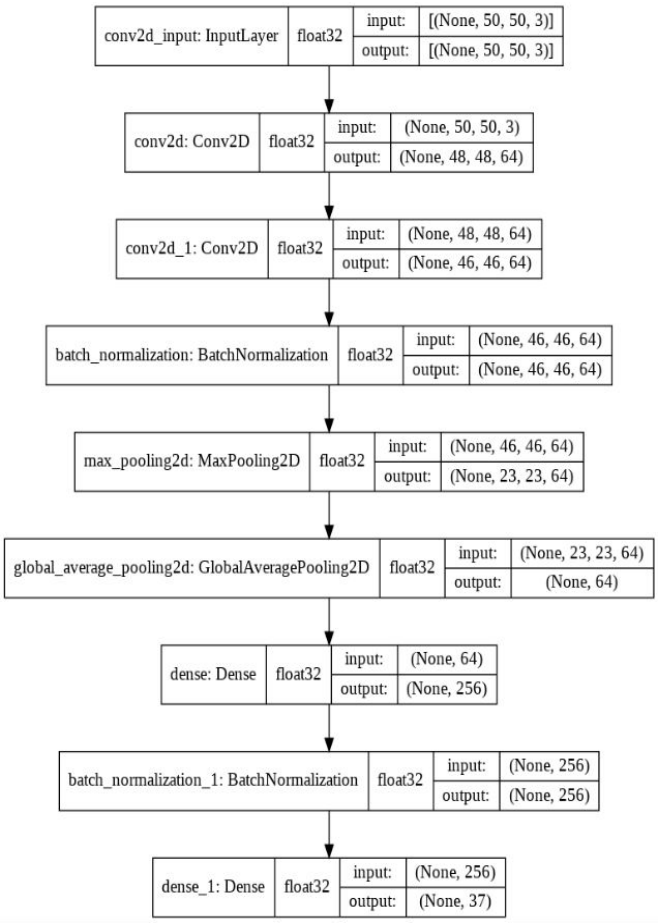


Fig. 13. Own Architecture Implementation

In our own architecture, we have implemented 3 convolution layers followed by batch normalization and max pooling, followed by global average pooling with dense layer and batch normalization, and a final dense layer for classification.

3.5 Proposed Model

We have trained 3 models with 2 different datasets to perform well on unseen datasets. We have trained LeNet-5, MobineNetV2 and, our own architectures. We have not taken the best model out of 3 we have taken all 3 models and made a final model that will perform an ensemble of these 3 models.

3.5.1 Neural Network Ensemble Horizontal Voting

In Machine Learning we have an ensemble technique where we train multiple sub-models and average them. Random Forest algorithm is an example where it uses multiple Decision tree algorithms.Similarly, we can perform ensemble for Neural Networks [4] as well. There are a lot of ensemble techniques for NeuralNetworks like Stacked generalization [8], Ensemble learning via negative correlation [9] and, Probabilistic Modelling with Neural Networks [10] [11]. We have implemented the Horizontal Voting Ensemble method to improve the performance of neural networks.

Horizontal voting is an ensemble technique for neural networks where we train several sub-models and make predictions using these sub-models.For the final predictions, we make predictions from all the sub-models and see which class has got maximum votes.The final prediction will be the class that has the maximum votes. For this, we have used 3 models that arean odd number of sub-models to avoid an even number of votes for two classes in worst cases.

Let model be the set of neural network models being trained on the training set $T(x_i,y_i)$, such that $m \in$ model. Let yhat be the predictions obtained by all the models on the test set $T'(x_i',y_i')$.

Let 'array' be the function for converting lists to arrays

Algorithm 1: Horizontal Voting

Input: models, test set $T'(x_i',y_i')$, and empty yhat list

Output: predictions – final prediction obtained

1. **Step 1:** Obtain the predictions of each model
2. **for** each i in range(x_i)
3. **foreach** m **in** model, **do**
   yhat[i] ←**predict**(x[i])
   Calculate highest number of votes for ith test data and append
   yhat[i] ← highest voted class
   endfor
4. **end for**

5. **Step 2:** Convert list into an array
6. yhat←**array**(yhat)
7. **Step 3: return** yhat

For MobileNetV2 model we have takenAdam optimizer with learning rate = 0.001,eta_1=0.9, beta_2=0.999, and epsilon=1e-07

For all the models while training we have used ReduceLROnPlateaucallback with factor = 0.2, patience = 2, min_lr = 0.001.

By using this horizontal ensemble technique, we have achieved 99.8% accuracy.

We have deployed all the models in Django Web Frameworks and built a simple frontend to accept the image from users and send the response.

We have also built an API that will pop up the live camera and detects the hand gestures and then converts them to the corresponding English alphabets.

4. EXPERIMENTAL RESULTS

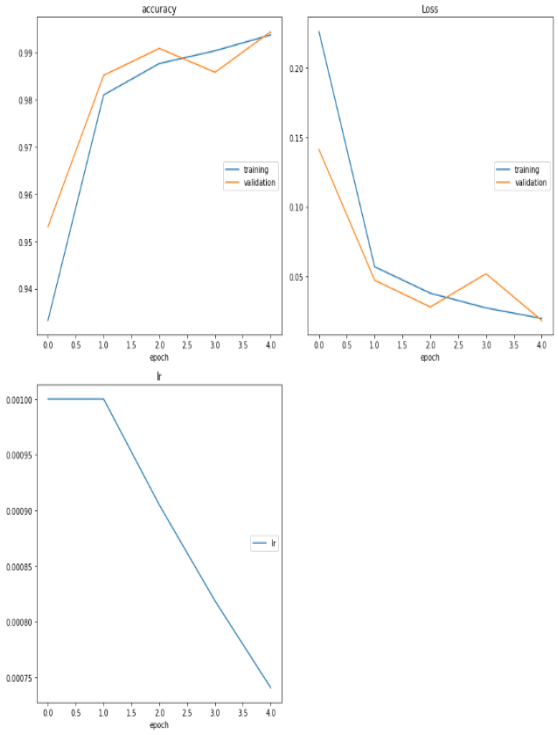


Fig. 14. Traininggraphs

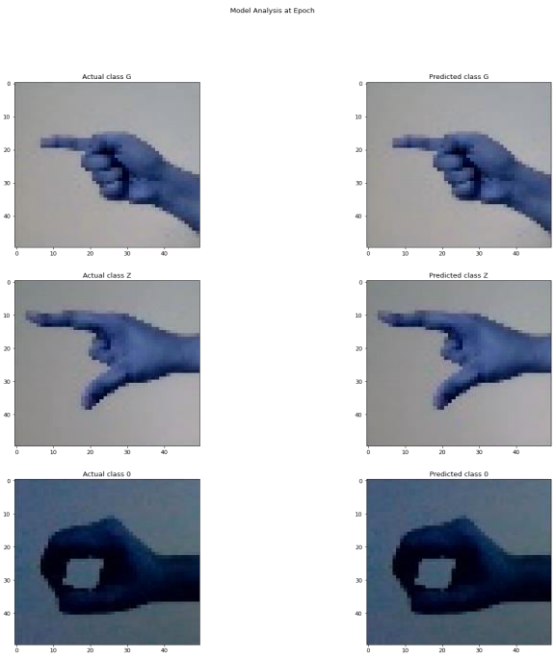


Fig.15. Model Prediction during training

Models	**Accuracy**
MobileNetV2	98.9%
LeNet-5	97%
Own Model	98%
Ensemble	**99.8%**

Table. 1. Performance Results

We have trained all the models for around 10-15 epochs with a batch size of 32.

All the models performed well on the test cases. After applying the horizontal voting ensemble technique for these 3 models, we have achieved almost 100% accuracy.

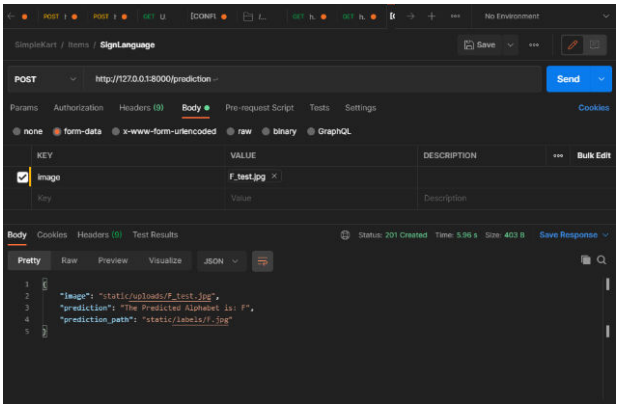


Fig. 16. JSON Response from the Application

We have used Django Rest Frameworks as a backend for our project. This is the sample JSON response that is sent to the frontend when a user inputs an Image. We have used OpenCV to test our results in the live camera. This is a sample result on a live camera

Fig. 17. Model Prediction on Live Camera Prediction: "W"

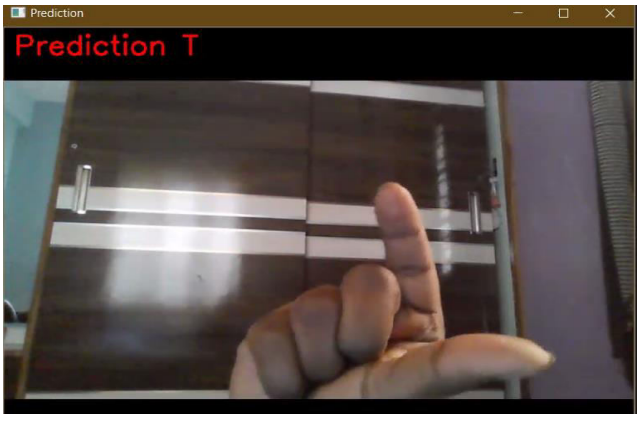


Fig. 18. Model Prediction on Live Camera Prediction: "T

Fig. 19. Model Prediction on Live Camera Prediction: "C"

Fig. 20. Model Prediction on Live Camera Prediction: "M"

5. CONCLUSION

In conclusion, we were successfully able to develop a practical and meaningful system that can able to understand sign language and translate that to the corresponding text. There are still many shortages of our system like this system can detect 0-9 digits and A-Z alphabets hand gestures but doesn't cover body gestures and other dynamic gestures. We are sure and it can be improved and optimized in the future.

REFERENCES

[1] Brill R. 1986. The Conference of Educational Administrators Serving the Deaf: A History. Washington, DC: Gallaudet University Press.

[2] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

[4] L. K. Hansen and P. Salamon, "Neural network ensembles," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993-1001, Oct. 1990, doi: 10.1109/34.58871.

[5]Kang, Byeongkeun, Subarna Tripathi, and Truong Q. Nguyen. ”Real- time sign language fingerspelling recognition using convolutional neural networks from depth map.” arXiv preprint arXiv: 1509.03001 (2015).

[6] Suganya, R., and T. Meeradevi. ”Design of a communication aid for phys- ically challenged.” In Electronics and Communication Systems (ICECS), 2015 2nd International Conference on, pp. 818-822. IEEE, 2015.

[7] Sruthi Upendran, Thamizharasi. A,” American Sign Language Interpreter System for Deaf and Dumb Individuals”, 2014 International Conference on Control, Instrumentation, Communication and Computa.

[8] David H. Wolpert, Stacked generalization, Neural Networks, Volume 5, Issue 2, 1992, Pages 241-259, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(05)80023-1.

[9] Y. Liu, X. Yao, Ensemble learning via negative correlation, Neural Networks, Volume 12, Issue 10,1999, Pages 1399-1404, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(99)00073-8.

[10] MacKay D.J.C. (1995) Developments in Probabilistic Modelling with Neural Networks — Ensemble Learning. In: Kappen B., Gielen S. (eds) Neural Networks: Artificial Intelligence and Industrial Applications. Springer, London. https://doi.org/10.1007/978-1-4471-3087-1_37

[11] Polikar R. (2012) Ensemble Learning. In: Zhang C., Ma Y. (eds) Ensemble Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-1-4419-9326-7_1

International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 5, Sept- Oct-2021, ISSN (Online): 2395-566X

Crop Infection Detection Using Yolo

[1]Satwik Ram Kodandaram, [2]Kushal Honnappa, [3]Parikshith H, [4]Sandesh S, [5]Kushal C

[1]AI/NLP Scientist, ISG (Information Services Group), Karnataka, India
[2] AI/NLP Scientist,Avkara/Vmitis, Karnataka, India
[3] 7th Sem UG Student,Visvesvaraya Technological University, Karnataka, India
[4] 7th Sem UG Student,Visvesvaraya Technological University, Karnataka, India
[5] 7th Sem UG Student,Visvesvaraya Technological University, Karnataka, India
E-mail: satwikram29@gmail.com,kushal.h1999@gmail.com, parikshithrao15@gmail.com, shivakumarsandesh2@gmail.com, kushalc658@gmail.com

Abstract-Agriculture is the backbone of a country. It is important to note that without agriculture, there is no economic growth in the country. As Technology has improved a lot and improving a lot day by day, these technologies can be utilized in farming and agriculture so that there will be maximum utilization of crops and less wastage of crops. To achieve this, we need to come across a few challenges. Which crops can be grown depending on certain weather conditions? Identification of disease in crops so that we can prevent it and maximum yield of crops. Prevention is better than cure the famous quote says. Artificial Intelligence is one of the greatest inventions, using AI we can train the machine with images to detect disease in crops. The problem of the underutilization of crops can be achieved. This paper proposes a model for implementingcropinfectiondetection and maximum yield of crops using Convolution Neural Networks (CNN) and You Look Only Once(YOLO).

Keywords-Agriculture; Crop disease; yield, image detection; CNN; YOLO.

I. INTRODUCTION

Agriculture is essential in the country's economic growth. The primary sector of an economy comprises agriculture and other activities which is significant to the Gross Domestic Product (GDP). For decades, agriculture has been associated with the productions of a maximum of crops, food, and other raw materials to the country. At present, agriculture includes farming, fruits, vegetables, dairy, mushrooms, etc.

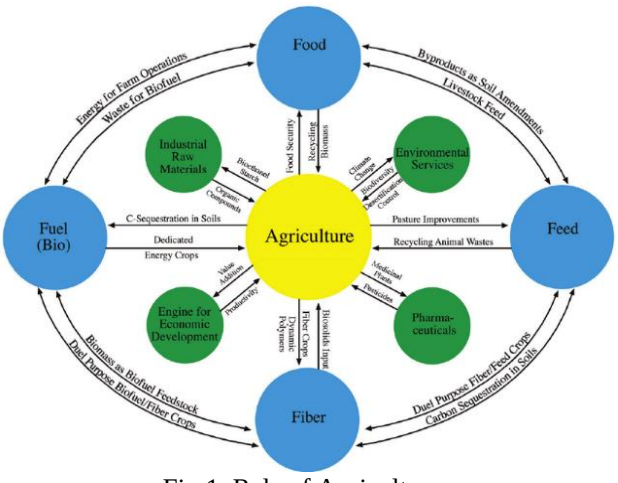


Fig 1. Role of Agriculture.

Agriculture not only provides food it also provides employment opportunities to a very large percentage of the population of a country. Crops are living plants grown by farmers. The majority of the crops are food-related, such as fruit, vegetables, and grains. Some crops are grown for fabrics like cotton, pharmaceuticals like quinine, and other products like wood and rubber.

Most of the crops are destroyed due to the rain or due to other climatic conditions. Most of the crops are grown and are affected by diseases, and these diseases are spread over the crops. If we overcome these challenges, we can achieve maximum yield and we can overcome the underutilization of crops.

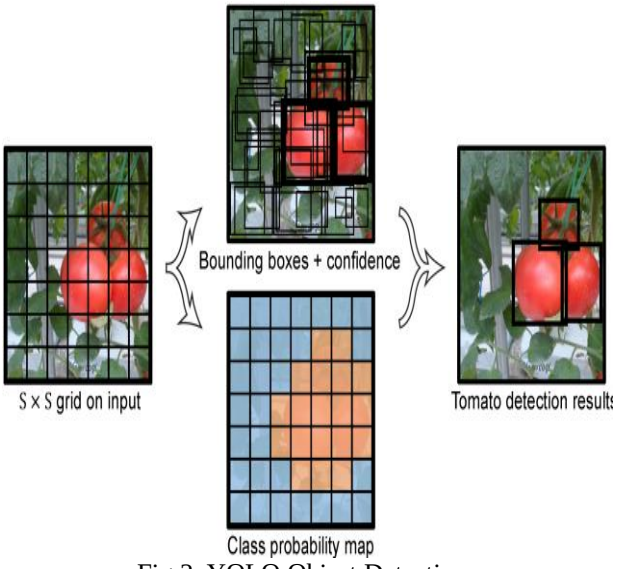


Fig 2. YOLO Object Detection.

Our proposed model provides the solution in the agriculture field. YOLO which was discovered by Joseph Redmon provides the best result in detecting the disease present in the crops.

A plant's leaf is the first part of the plant which shows symptoms of diseases affecting the plant. Object detection using the YOLO model is applied to plants leaf to detect disease that occurred in the plant at an earlier stage before itsaffects the entire plant. Figure 2 demonstrates how YOLO works in the objection detection of a plant.

Using this YOLO Object detection we will spot all the infected places in a leaf and put a bounding box. If a leaf is healthy the same bounding box will be put indicating the leaf is healthy.

Fig 3. Healthy Leaf

Fig 4. Bacteria Infected Leaf

1.Motivation:
The main motive of our proposed modes is as follows.

- Identifies which area of the plant/leaf is affected and preventive measurements can be taken care of.
- Predicting the number of defects in individual plants to understand
- This application will be installed in the farmland once the occurrence of the disease is found necessary pesticides can be applied to the affected area.
- Identifying the infected plant manually consumes more time and also error in identifying them is more. This also requires more labor, Using our proposed model sorts all these issues.

2. Problem Statement:
Farmers have been able to identify crop diseases with their naked eye since the beginning of time, and they continue to do so today, which forces them to make difficult decisions about which fertilizers to employ. It necessitates a thorough understanding of disease kinds as well as a great deal of expertise to ensure accurate disease identification. Farmers are frequently perplexed by diseases that appear to be almost identical. The below image showsmore details.

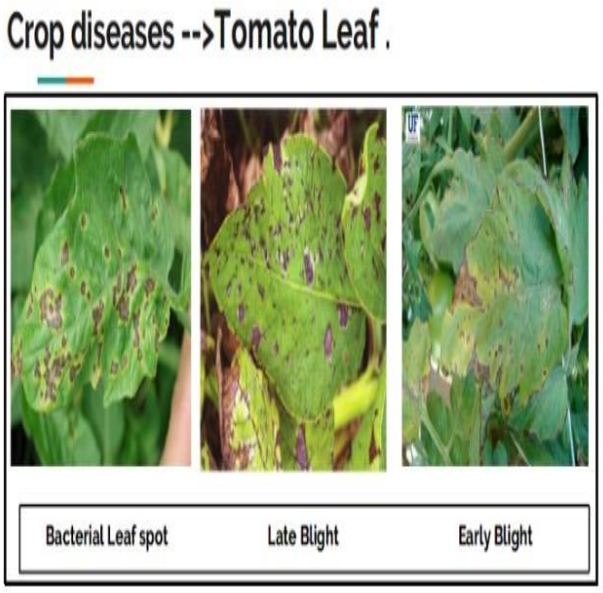


Fig 5. Crop Diseases.

In Today's agriculture, the crops grown by the farmers are underutilized. Weather conditions fluctuate a lot. Most of the time crops are damaged by rainfall. The following are the few problems in today's agriculture:

- Crops are damaged due to climatic changes such as rainfall, temperature, etc.
- Wrong seasonal crops are grown.
- Crop diseases and the spreading of these diseases to other crops.

- Wastage of crops (Underutilization) due to diseases of crops.
- Less yield of crops.

The leaf in the above fig 5, all are similar and it is very important to identify which plant leaf it and which fertilizer should be used to avoid crop diseases.To avoid this, excellent recommendations on which fertilizer to use, accurate diagnosis of crop bacterial infection, and the capacity to distinguish between two or more interdependent types of diseases in visuals by adding bounding boxes are all required.

3. Objectives:

The main objective of our proposed model is to help farmers in the prevention of crop diseases at an earlier stage by using deep learning techniques such as objection detection, computer vision. The occurrence of disease in a plant leaf is detected. All the affected area is identified using the bonded box. By this farmer can apply necessary medicines to that area or remove that part of the plant before it affects the entire plant.Building end-to-end application where YOLO is used for object detection.

The plant village dataset is trained using Convolution Neural Network for several iterations this trained model is used for deployment using the Django framework. This will provide a user interface for the farmers by which they can identify the occurrence of disease in the plants. Then they can spray pesticides on these plants and protect the plant before the whole plant is affected.

II. LITERATURE REVIEW

There are many systems developed to detect disease in crops and to achieve maximum yield. But the performance and stability problems.In [1], Wang R et al. proposed a system that's based on the detection of diseases in plants using Image processing Technology using Mat lab. This system works well in but when it comes to practical implementation such as direct detection using cameras, this system may fail to give accuracy, where we can achieve using python (YOLO).

In [2], Dengshan et al. proposed a system that's based on Crop disease Detection using Image Segmentation which uses Image processing and K-means clustering Technique. The problem might be the initialization of K clusters. And K-Means clustering technique has been applied to solve low-level image segmentation tasks.

In [3], Ferentinos et at. have used convolution neural network to identify plant disease they have trained for 58 different classed dataset that consists of 25 different plants. this system achieved 99.53% accuracy in making the prediction this was achieved by using VGG architecture.

III. IMPLEMENTATION

1. Dataset:

The data set consists of 54000 leaf images of various plants. This dataset consists of images of both healthy and diseased plants images. The healthy category is to make the model understand which plants are disease-free and keep a track of them. In the disease category, there are more than one disease labels based on the kinds of diseases that the particular plant gets affected.

We have trained for around 38 distinguish classes. This category includes most of the plants commonly grown by farmers. A diseased category is identified if a small area of the leaf is defective and want kind of disease caused that disease.A small snippet of our dataset is shown in figure fig 6. Here we can see that the labeling of a class is given by plant name followed by the disease it is affected if any else it's labeled by plant name followed by healthy as its category.

Fig 6. Crop Diseases Dataset

The dataset includes plant image such as

- Strawberry
- Tomato
- Soybean
- Potato
- Corn
- Orange
- Blueberry
- Cherry
- Grapes
- Peach
- Pepper
- Raspberry
- Squash
- Apple

There are 14 different plants in total. The kinds of diseases caused to them are viral disease, bacterial diseases, disease due to mite, common diseases, and diseases caused by mold.

2.ImageAnnotation:

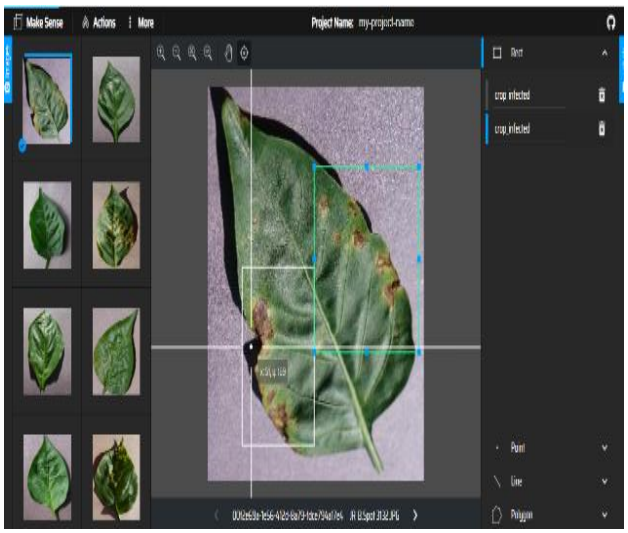


Fig 7. Image Annotation

Image annotation is one of the most important tasks in Computer Vision. Image annotation is the human-powered task of annotating the images with labels. Here annotating in the sense, we are manually putting the bounding boxes on the images and labelingthem using annotating tools which are available online.

There isa lot of annotation format. For YOLO .txt format is used with the same name of the image file in the same directory thefile will be created.
Each .txt file contains
<object-class><x><y><width><height>
Example:
0 45 55 29 67
1 99 83 28 44

3. Convolution Neural Networks (CNN):

The Convolution Neural Network is an algorithm that is particularly constructed for pickle-related data. This is a deep learning algorithm generally used for processing and recognizing image-related datasets.

Unlike other classification algorithms, CNN requires less pre-processing of the data. This algorithm was inspired by neurons in the human brain particularly called as Visual Cortex which is present in the occipital lobe and this is responsible for visual information.

The image acquired is converted into pixel data which is on processing according to CNN architecture will be fit into Artificial Neural Network architecture. ANN will process and produce/predicts the output. YOLO Object Detection uses the CNN algorithm.

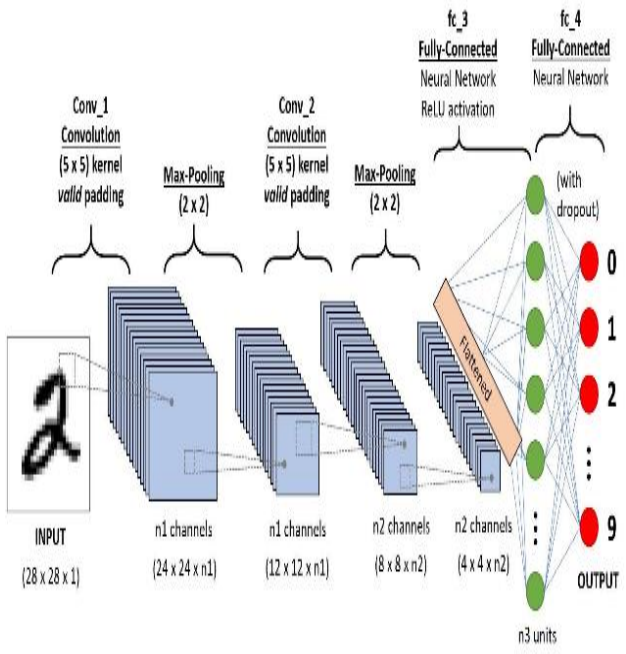


Fig 8. Working of Convolution Neural Network

4. Object Detection:

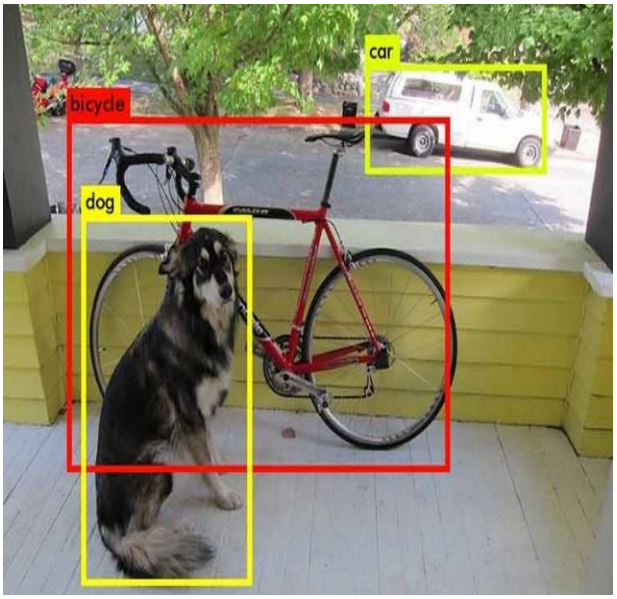


Fig 9. Object Detection

Object detection is the task of detecting the instances of objects of a certain class. There are many Object detection algorithms, You Only Look Once (YOLO) is one of them. YOLO uses CNN in the backend and learns from the annotated images.

5. You Only Look Once (YOLO):

"You Only Look Once" (YOLO) is one of the most popular object detection algorithms used for real-time object detection tasks. YOLO has been widely used since its first release.

1. Previous YOLO Versions

1. YOLOv1:
YOLOv1 was first introduced in May 2016 by Joseph Redmon with paper [4]. This was one of the biggest evolutions in object detection.

2. YOLOv2:
In December 2017, JosephRedmon introduced another version of YOLO with paper [5] and it was known as YOLO 9000

3. YOLOv3:
After a year release of YOLOv2 in April 2018, the most stable version of YOLO was introduced and it was released with paper YOLOv3 [6]

4. YOLOv4:
Finally, in April 2020, Alexey Bochkovskiy introduced YOLOv4 with paper [7]. YOLOv4 was introduced with some amazing new features, it outperformed the previous version YOLOv3 with a high margin and also has a significant amount of average precision when compared to EfficientDet Family.

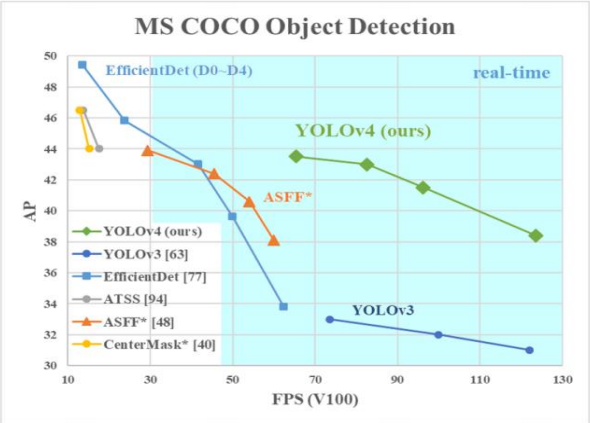


Fig 10. YOLOv4 Improvements

IV. YOLOV4 WORKING

YOLO takes an entire image in a single shot and predicts the bounding box coordinates and class probabilities for these boxes. The Biggest advantage of YOLO is its lightning speed of execution. It's incredibly fast in execution and can process up to 45 frames per second.

YOLO also understands generalized object representation. The network does not look at the entire image, only at the parts of the images which have a higher chance (probabilities) of containing an object.

YOLO first takes input images and then divides the images into grides.

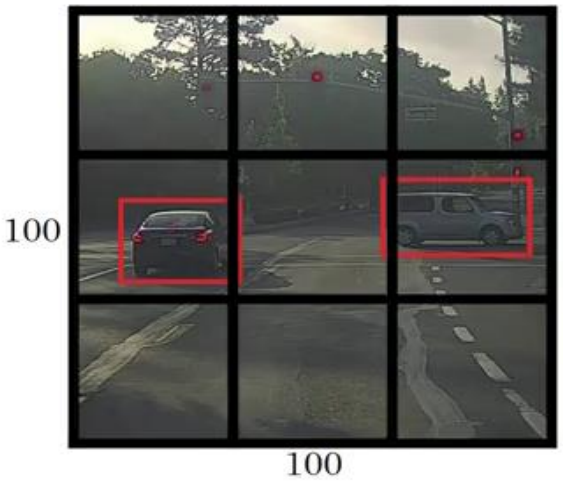


Fig 11. Image Divided into grides

For each box, it checks whether an image is present or not. It calculatesthe following:

1. X – Coordinate of the bounding box center inside the cell (0;1with respect to grid cell size)
2. Y – Coordinate of the bounding box center inside the cell (0; 1with respect to grid cell size)
3. W – Bounding box width [0;1] with respect to the image.
4. H – Bounding box height [0;1] with respect to the image.
5. C – Bounding box confidence(object in the box)

PC
C1
C2
C3
X
Y
W
H

PC represents whether an object present in the grid or not (Probability)

X, Y, H, andW represents the bounding box of an object
C1, C2, and C3 represent the class of objects say car, bike, bus, etc
If PC = 0 then C1, C2, C2 = 0
If PC = 1 it tries to find the centre of the object available.
If it finds multiple bounding boxes it uses Intersection Over Union (IOU) to find the best bounding box.

$$\mathbf{IOU} = \frac{\textbf{Intersection region}}{\textit{Totalareaoftheboxes}} \textbf{(1)}$$

If IOU > 0.5 then it takes the highest probability.

Once we get the best bounding box of which object exits it uses CNN to classify the images.
This is the basic working of YOLOv3.

Most people and Scientists in the field today use YOLOv3, which is already producing excellent results. YOLOv4 has improved again in terms of accuracy and speed (Frame Per Second), the two metrics we generally use to evaluate an object detection algorithm.

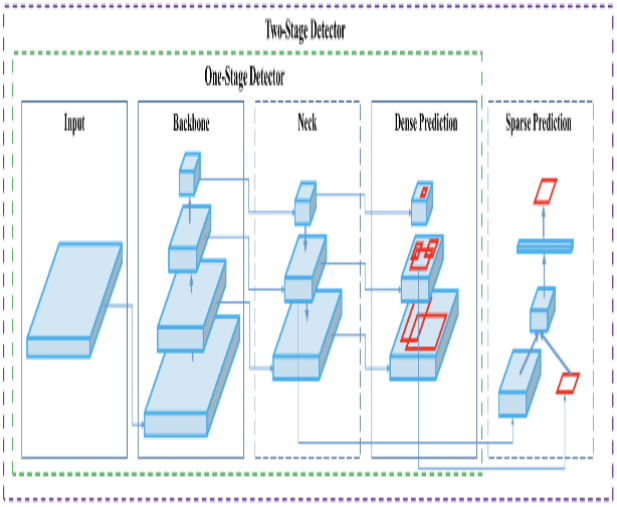


Fig12.YOLOv4 Architecture.

In YOLOv4 we have 3main blocks after the input image is processed:

- Backbone
- Neck
- Head(Dense Prediction)

	Type	Filters	Size	Output
	Convolutional	32	3 x 3	256 x 256
	Convolutional	64	3 x 3 / 2	128 x 128
1x	Convolutional	32	1 x 1	
	Convolutional	64	3 x 3	
	Residual			128 x 128
	Convolutional	128	3 x 3 / 2	64 x 64
2x	Convolutional	64	1 x 1	
	Convolutional	128	3 x 3	
	Residual			64 x 64
	Convolutional	256	3 x 3 / 2	32 x 32
8x	Convolutional	128	1 x 1	
	Convolutional	256	3 x 3	
	Residual			32 x 32
	Convolutional	512	3 x 3 / 2	16 x 16
8x	Convolutional	256	1 x 1	
	Convolutional	512	3 x 3	
	Residual			16 x 16
	Convolutional	1024	3 x 3 / 2	8 x 8
4x	Convolutional	512	1 x 1	
	Convolutional	1024	3 x 3	
	Residual			8 x 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Fig13.YOLOv4 Backbone Darknet-53

1. Backbone:
Backbone here refers to the feature-extraction-architecture. Here the backbone can be EfficientNet, ResNeXt,ResNet, Darknet53,SpineNet, or VGG. In the official YOLOv4 paper they have used CSPDarknet53 architecture.

2. Neck:
The main purpose of the neck block is to add extra layers between the backbone and the head (dense prediction) block.

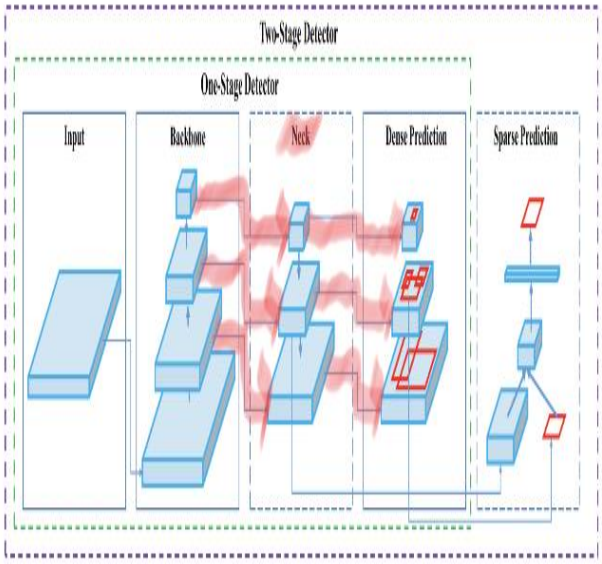


Fig 14.Different feature maps used in YOLOv4.

3. Head (Dense Prediction):
The head block has two parts:

- Locate bounding boxes.
- Classify the image inside each box.

This process isthe same ashow YOLOv3 works. The network detects (x, y, w, h) and a confidence score for a class.

V. PROPOSED MODEL

In our proposed model we aim to identify the bacterially infected part of the leaf. If there is any infected part, we identify it by putting a bounding box on it with the disease name.We have annotated the images using an online tool and we have trained the YOLOv4 for about 6000 to 10000 epochs for 1000 images per class.We have trained with 14 different varieties of the crops and their diseases with YOLOv4. If the model identifies the diseases and bacterial infected part, the model will suggest some of the fertilizers which can be used to improve the quality of the crops and hence increase yield.

If there is no bacterial infected part then the same model will put a bounding box saying it's healthy.We can use this

model on live cameras and able to get some recommendations regarding crop disease and fertilizers to improve the quality of the crops.

It is often very difficult to identify the type of crop disease by looking at the leaf. And we have to be very careful in identifying the type of fertilizers for each crop. Using suitable fertilizers for the crops helps farmers to get a good yield for the year.

VI. EXPERIMENTAL RESULTS

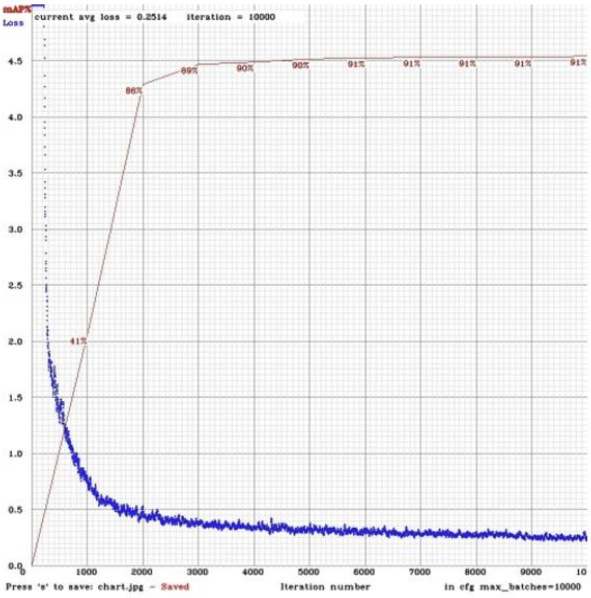


Fig 15.Training plots

Fig 15 shows the training graphs which was plotted during model training. It is a graph between Loss vs Epochs. Here the model is trained around 10000 epochs.

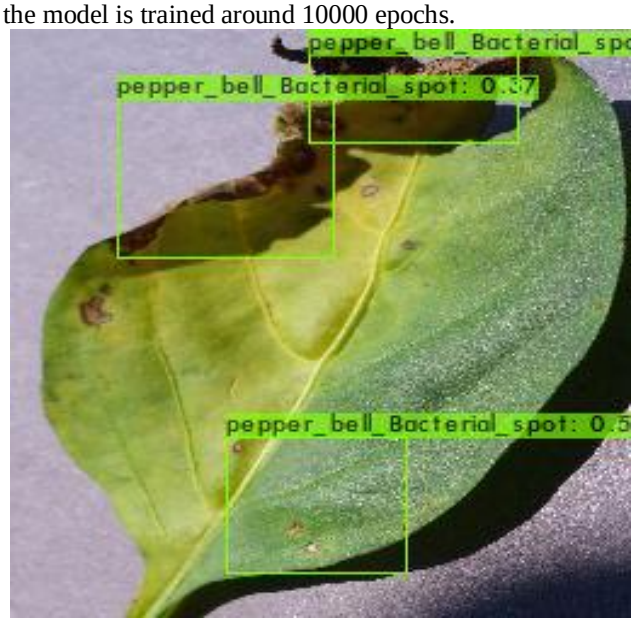


Fig 16. Model Prediction – Infected Spot.

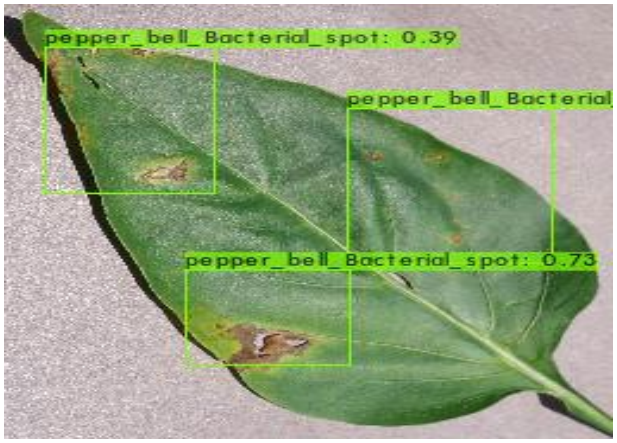


Fig 17. Model Prediction – Multiple Infected Spots.

We can see our proposed model identified multiple bacterial infected spots. The model has trained about 10000 epochs and we got amazing results on test data. Upon increasing the number of images per class and epochsthese results can be still improved with better bounding box confidence. Once we get this output from the model with the bounding box and its crop type, we can able to suggest some fertilizers which can be used to overcome the infection.

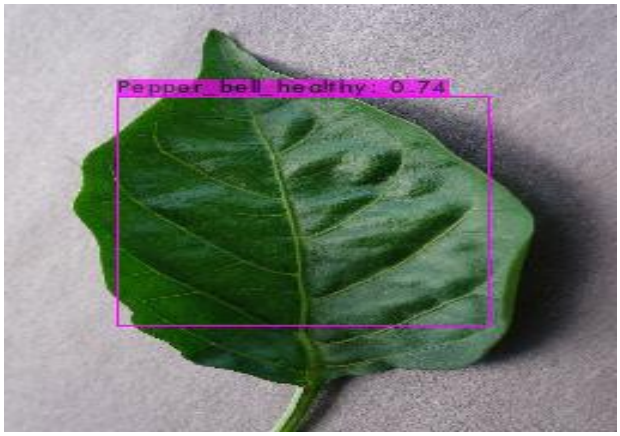


Fig 18. Model Prediction – Pepper Bell Healthy.

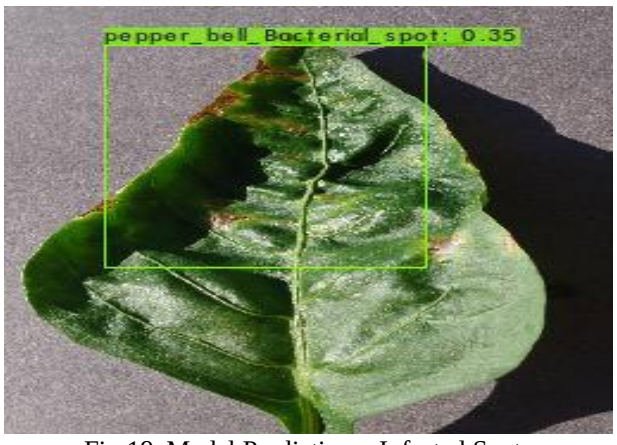


Fig 19. Model Prediction – Infected Spot.

Fig 20. Model Prediction – Pepper Bell Healthy.

VII. CONCLUSION

YOLOv4 an object detection algorithm was used to identify whether the plant is healthy or infected by any diseases, suppose the plant is affected then the model will identify what kind of disease that the plant is affected and our application will recommend the farmers which pesticides to be sprayed or recommends feasible solution in recovering from that disease. This helps farmers in increasing the production of the crop and decreases the labour force which helps to get better profit in the market.

REFERENCES

[1] Li D, Wang R, Xie C, Liu L, Zhang J, Li R, Wang F, Zhou M, Liu W. A Recognition Method for Rice Plant Diseases and Pests Video Detection Based on Deep Convolutional Neural Network. Sensors. 2020; 20(3):578. https://doi.org/10.3390/s20030578.

[2] Li, Dengshan; Wang, Rujing; Xie, Chengjun; Liu, Liu; Zhang, Jie; Li, Rui; Wang, Fangyuan; Zhou, Man; Liu, Wancai. 2020. "A Recognition Method for Rice Plant Diseases and Pests Video Detection Based on Deep Convolutional Neural Network" Sensors 20, no. 3: 578. https://doi.org/10.3390/s20030578

[3] Ferentinos KP. Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture. 2018 Feb 1;145:311-8.

[4] Redmon, Joseph, S. Divvala, Ross B. Girshick and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 779-788.

[5] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.

[6] Redmon, Joseph and Ali Farhadi. “YOLOv3: An Incremental Improvement.” ArXiv abs/1804.02767 (2018): n. pag.

[7] Bochkovskiy, Alexey & Wang, Chien-Yao & Liao, Hong-yuan. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.

Reverse Recommendation: without user details

[1]Satwik Ram Kodandaram, [2]Kushal Honnappa and [3]Kunal Soni

[1]AI/NLP Scientist, Avkara Technologies, Banswara, India
[2] AI/NLP Scientist, Avkara Technologies, Banswara, India
[3]CEO, Avkara Technologies, Banswara, India
E-mail:satwik.ram@vmitis.com, kushal@vmitis.com,kunalsoni@vmitis.com

Abstract

The recommendation is the new and probably the only method to ensure any online application effectively caters to a global audience. An application that caters to a very wide group of people from different backgrounds so people can find the data they could be looking for. We see the use of recommendations almost everywhere in our digital lives. We can look at recommended apps, posts, dishes, songs, and this list extends till infinity. Personalizing an application for any specific user helps in achieving better conversion rates. This is achieved through recommending the user based on the data collected about the user. The more data is collected, the more precise the recommendation gets, thus resulting in greater conversion rates. Various machine learning algorithms could be used for creating a recommendation system based on the type of recommendation required. The recommendation could get better with the collection of more data and then training the algorithm on the particular dataset. More precise recommendations could lead to the reverse effect of the recommendation. It could be used to model the user behavior according to what the developer wants. This could lead to negative consumer patterns as user now does what the developer wants them to do. But this does not mean we forbid the idea of recommendation asa whole. A recommendation is a great tool for achieving more conversion rates. In this paper, the authors propose recommending using groups. This not only ensures that reverse consumer behavior is prevented from being achieved, but is also computationally cheaper.

1 Introduction

The recommendation comes with a big challenge to user privacy. The recommendation is based on the user's activity, so there is a possible chance that a user's behavior is recommended to some other user. Individual user tracking can also bring issues of privacy. In this paper, the authors use sentiment analysis to group data and flag a user in a particular category. The flag can change as per the user activity throughout the application.

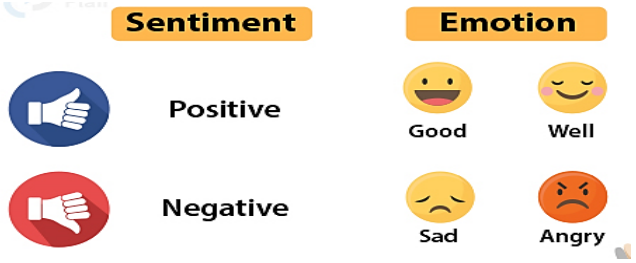


Fig 1. Users Emotions

This way, the users are not tracked individually but are flagged under few categories and are recommended data according to the flag. The recommendation could not be avoided in an application that is catering to a very wide group audience but there could be other ways to recommend that retain the user privacy as well. With this implementation, as there is no tracking, authors expect more expression from the users as they now enjoy the privacy

1.1 Motivation

The recommendation has its negative effects on the users and most of the users are not aware of these effects that could affect them. Even if the users are aware, they don't have any choice. To recommend the system needs user's data that includes their activity their browsing history. All these data will be trained whichaffect the privacy of the user, but Group recommendation will not target any user particularly, this giving space for privacy and even reducing computational costs. Here the group recommendation will be done based on the sentiment of user's posts that are posted in the application.

1.2 Problem Statement

An individual recommendation can lead to the problem of privacy. Authors believe that this privacy issue can limit the emotions that are expressed by people. Also, more recommendations can reverse the effect of recommendation. This tool can be used to psychologically change the behavior of the people to make choices that the developer wants them to. This is a negative thing as it violates the very spirit of consumerism. The main disadvantage is as we are not collecting any user's information. That makes it difficult to recommend individuals based on their identity.

1.3 Objectives

The objective is to flag a user under a specific category as per their interaction with the application and then recommend data from the particular category. The user's category could change from time to time based on the changed category. The recommended posts will be recommended based on sentimental analysis of the created data which will then be recommended to all the users in the particular category. The main objective is to make a depressed or mentally ill person come out of it and keep entertained the happier person and make them more strong than before.

2 LITERATURE REVIEW:

We can find traces of notable work done in recommending using grouping. Though the application of the existence of such a system is different from the method proposed by the authors:

In [1] James et al. discuss YouTube's video recommendation system that recommends videos based on user's activities over the site these include users liked, watched, and favorite videos. They recommend populated videos for the user who just wants entertainment by the content.

The recommendation is updated regularly bases on the user's recent activity. Following that, the videos are evaluated for relevance and diversity using a variety of signals. This system aims to provide privacy. The main metric that is considered is click-through rate (CTR) This is evaluated based on time until first recommendation coverage, session length, and long watch. These parameters are used to assess system overall performance as well as to evaluate system changes in real-time traffic. They consider top viewed, the most added as a favorite by most users, and rates videos receiving the most rating this observed for 21 days.

In [2] Joonseok et al. explain the implementation of a recommendation system in an academic paper that reduces the load on researchers in searching the papers based on keyword-based or browsing papers based on top journals or conferences. They use computational evaluation and user study demonstrate techniques to build recommendation systems that recommend users interesting papers.

This is done by applying the bag-of-words model to the corpus and learning is done using a lazy method similar to kNN. The preference of the user is determined and relatively paper is recommended. Here neighbor-based and clustering recommendation algorithm is applied. The results are transferred by the visualizer to users.

In [3] Tetsuya et al. illustrate the sentimental analysis of a particular subject from a document instead of analyzing the whole document is positively polarised or negatively polarised. The main target is to identify whether a statement is positive (favorable) or negative (unfavorable) to achieve this they have used these steps that are Expressions of emotion, their polarity and power, and their connection to the subject.

It is said that all these are correlated. In this entire process, Natural Language Processing is used to understand the sentimental part of speech (POS). the achieved 95% precision and round 20% recall during the initial stage, it is said based on the observation that the precision may go down till 75 %. In a feature, they are working on the automated generation of the sentiment lexicons to reduce human effort and also a new way to improve

In [4] Amer et al. talk about the importance of grouping to recommend data to a group of people together. For example, recommending a movie to a group of friends watching it together.

In [5] Kim et al. talk about the working of a recommendation system that works in two stages to recommend data to a community of people.

In [6] HaiYan et al. talk about dynamically recommending by adjusting weights of the data points dynamically.

3. Methodology

3.1 Sentimental Analysis

The process of detecting positive or negative sentiment in the text is known as sentiment analysis. Sentiment analysis is the systematic identification, extraction, quantification, and study of affective states and subjective information using natural language processing, text

analysis, computational linguistics, and biometrics. Businesses frequently utilize it to detect sentiment in social data, assess brand reputation, and gain a better understanding of their customers. Sentimental analysis is required in every area where users feedback is applicable. The sentimental analysis system analysis the feedback given by the user based on its business leaders can understand how much the users are satisfied with their products. Based on the number of negative feedback provided by the users the company can improve the product.

Fig 2. Sentimental Analysis

Sentiment analysis is becoming a crucial tool for monitoring and understanding client sentiment as they share their opinions and feelings more openly than ever before. Brands can learn what makes customers happy or frustrated by automatically evaluating customer feedback, such as comments in survey replies and social media dialogues. This allows them to customize products and services to match their customers' demands.

3.2 Recommendation System

Recommender systems are computer programs that make recommendations to users based on a variety of parameters. These systems predict which products users are most likely to and are most interested in. Netflix, Amazon, and other similar companies are examples of this.

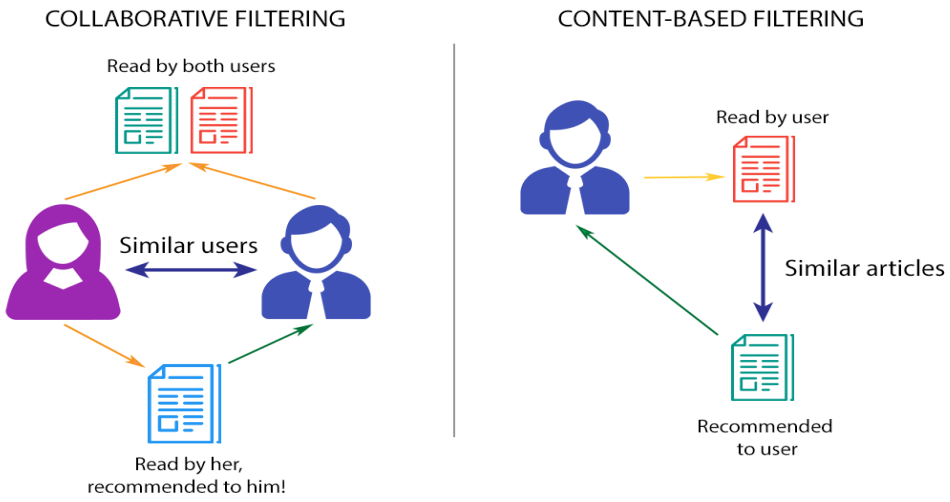


Fig 3. Recommendation Systems Methods

Here, the recommendation is based on user activity and most likely the recommendation will be of user interest and similar one. For example, if a user watches a video of the genre "horror" the system will recommend "horror" videos.

4. Proposed Model

Our application "Vmitis" focus is mainly on mental health, where people can share their taught and share their views irrespective of the subject without any identity that is our application has complete anonymity.

When a user is posting a "happy" article usually recommendation system will recommend the same "tag" article that is "happy" articles. This is completely acceptable. But when a person is in depression and he/she is posting more "depressed" or "sad" articles the system would recommend the same "tag" article that is "sad" articles. In this case, it is not acceptable. We mainly focus on people's mental health and we don't want to recommend "sad" articles and make people more depressed. Here usual recommendation system fails.

Our proposed model solves this issue. In our application, we have up to 3 tags for all the articles. When a user posts an article, we analyze the post using sentiment analysis. If we found the article to be "sad" we take the tag of the post and we recommend "happy" or any other tag articles. This model breaks the usual recommendation chain and comes out of the loop and recommends something out of the box. By this, we can improve the user personal feed, and also people can come out from "sad" or similar articles recommendations.

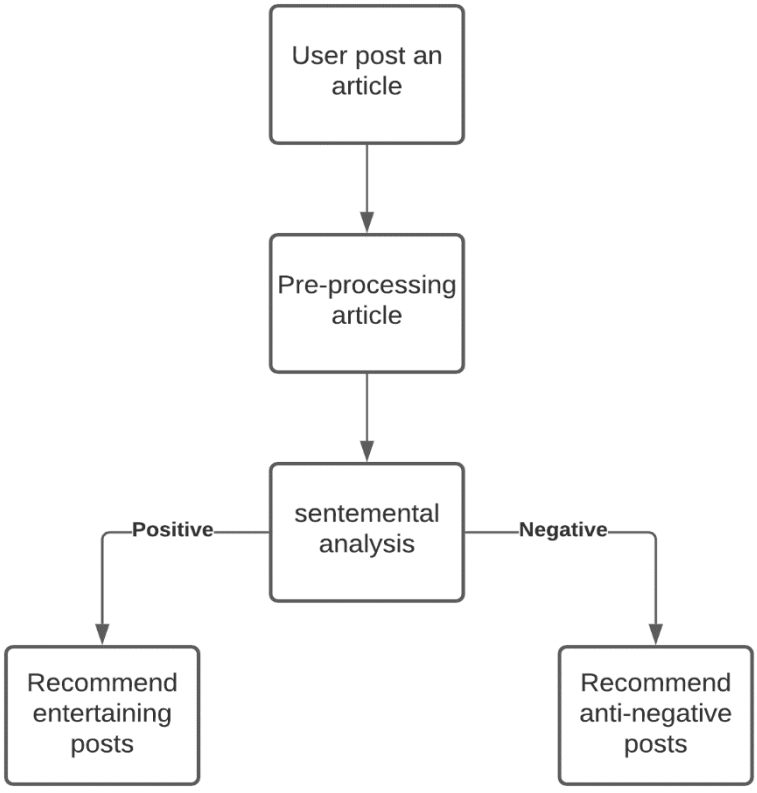


Fig 4. Proposed Method

In our application, we group users based on certain criteria and analyze the articles. Based on the certain polarity of the article and other patterns, we recommend the articles to the users.

Results

The model was trained with huge data which consists of both positively polarised and negatively polarized data. As a result of the trained model, the model made an accurate prediction in analyzing the sentiment of the article passed to it. Based on the sentimental the recommendation system starts to recommend the articles and post to the user. If the user is posting negative(depressed) data then the recommendation system will be recommending anti-negative articles to them to get them back out of negativity. Similarly is the user is posting a positive article then the user will be recommended a positive or entertainment post to engage them to be more stronger and helps him to be more active than before.

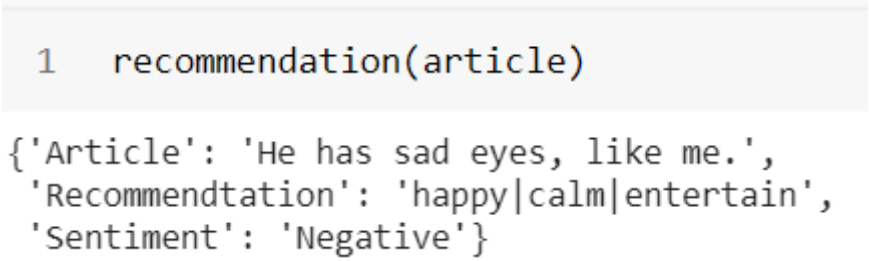


Fig 5. Negative article post detected: The customized function is written which detects that the article is positively polarised or negatively polarized. Based on it the recommendation will be recommended as seen in the above image

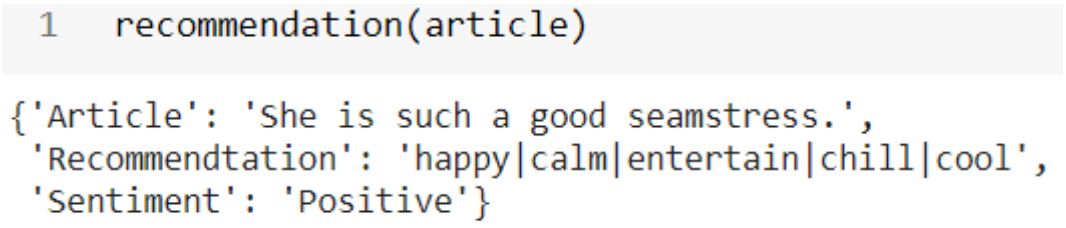


Fig 6. Positive article post detected: The customized function is written which detects that the article is positively polarised or negatively polarized. Based on it the recommendation will be recommended as seen in the above image

Conclusion

In this paper, we talked about flagging users under a category based on their interaction with the system. This flag will change as the user interacts with the system. This system will ensure that the users are not aggressively targeted for recommending. The authors talked about how the recommendation is a need for any application that is targeting a global audience for achieving more conversions. The data is categorized based on sentimental analysis. This categorization can later be extended to languages as well. Both these factors can later be mixed with different weights to create a more diverse recommendation system.

We should be aware that there should be control on the recommendation so there aren't any negative effects to it.

References

[1] Davidson, James, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta et al. "The YouTube video recommendation system." In *Proceedings of the fourth ACM conference on Recommender systems*, pp. 293-296. 2010.

[2] Lee, Joonseok, Kisung Lee, Jennifer G. Kim, and Sookyung Kim. "Personalized academic paper recommendation system." *SRS'15* (2015).

[3] Nasukawa, T. and Yi, J., 2003, October. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on Knowledge capture* (pp. 70-77).

[4] Amer-Yahia, Sihem, SenjutiBasu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. "Group recommendation: Semantics and efficiency." *Proceedings of the VLDB Endowment* 2, no. 1 (2009): 754-765.

[5] Kim, Jae Kyeong, HyeaKyeong Kim, Hee Young Oh, and Young U. Ryu. "A group recommendation system for online communities." *International Journal of Information Management* 30, no. 3 (2010): 212-219.

[6] Xu, Haiyan, Yanhui Ding, Jing Sun, Kun Zhao, and Yuanjian Chen. "Dynamic Group Recommendation Based on the Attention Mechanism." *Future Internet* 11, no. 9 (2019): 198.

International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 5, Sept- Oct-2021, ISSN (Online): 2395-566X

Improving the Performance of Neural Networks

Satwik Ram Kodandaram
AI/NLP Scientist
Information Services Group,
Karnataka, India
E-mail: satwikram29@gmail.com

Abstract- Deep Learning is a sub-part of Machine Learning where we exactly mimic the human brain neural network system. Deep Learning models are nonlinear models. They offer increased flexibility and can scale in proportion to the available training dataset. The downside of this flexibility is weights are calculated and updated via a stochastic training algorithm which means that they are sensitive to the training data and may have a different set of weights upon each time they are trained and produce different predictions. Generally, this case is referred to as neural networks with high variance and it will be very difficult to produce a final model for predictions. Deep Learning models often take too much time to train which means we require high computation resources like GPU or TPU. After investing so much time and resources, there is no guarantee that the final model will have low generalization error when performing on the unseen dataset. To overcome this, we need to reduce the variance of the model. A successful approach to reduce the variance is to go for "ensemble learning". In this paper, we will discuss different methods of "ensemble learning" to improve the accuracy of the deep learning model by reducing the variance.

Keywords- High Variance, Low Bias, Low Generalization, Reduce Variance, Ensemble Learning.

I. INTRODUCTION

The concept of "ensemble" is common in machine learning. Different algorithms are combined for single predictions. The most famous Random Forest algorithm is an example where it combines various multiple decision trees to build a model.

For Deep Learning also we can apply some ensemble techniques [1] [2] to improve the accuracy of the model. An increase in a few percent of the accuracy matters a lot when we have a large dataset.

For Deep Neural Networks we can improve the accuracy by changing the architecture of the model and by hyperparameters tuning that is by increasing/decreasing the number of hidden layers, changing weight initialization, changing activation functions, and trying with different combinations of it and picking the best hyperparameters that perform well on unseen data.

However, trying out different hyperparameters requires a lot of time as the model should be trained with all permutations and combinations. And hyperparameter tuning not always gives the best result because tuning the model also requires good knowledge of the subject. To overcome this, we need to reduce the variance of the model.

A successful approach to reduce the variance is to train multiple sub-models instead of a single model and combine all of them to predict a single prediction. This is called ensemble learning and this not only reduces the variance of the predictions but also can result in predictions that are better than a single model prediction. Different "ensemble" techniques can be implemented on Deep Neural Networks and improve the performance of the model.

The advantages of using ensemble learning are as follows:

- Improved Performance
- Improved endurance
- Can Build Robust Model
- Better Generalization compared to Single model

II. ENSEMBLE LEARNING

Suppose we ask a complex question to thousands of random people, and then aggregate their answers. In many cases, we will find that this aggregated answer is better than the expert's answer. This is called "Wisdom of Crowd".

Similarly, if we aggregate the predictions of a group of predictors such as classifier or regressor, we will often get better predictions than the individual best predictor. A group of predictors is called ensemble; thus, this technique is called Ensemble Learning, and an Ensemble Learning algorithm is called as Ensemble method.

An example of Ensemble learning in Machine Learning is the Random Forest algorithm in which trains a group of

Decision Tree Classifiers, each on a different random subset of training set.

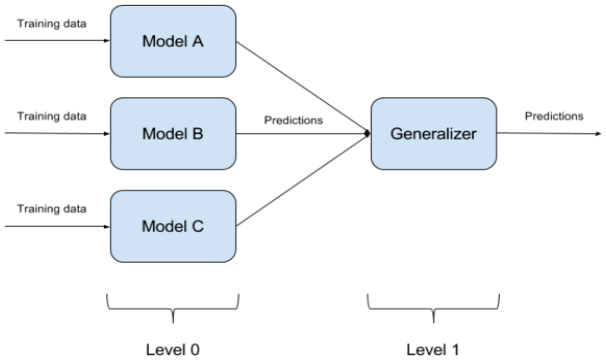


Fig 1. Ensemble Learning.

In Deep Learning we have different ensemble methods that can be implemented to improve the performance of the neural networks. Some of them are discussed in this paper.

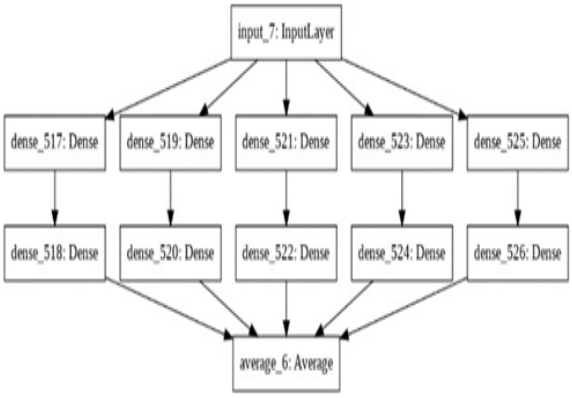


Fig 2. Neural Networks.

III. STACKING GENERALIZATION ENSEMBLE FOR DEEP LEARNING

A model averaging is an ensemble technique where it combines the prediction of multiple trained sub- models to a single prediction. Here all the sub-model is contributed equally to the combined single prediction. A limitation of this approach is here we are just combining the predictions of each sub- model regardless of its performance.

A slight variation of this version that is averaging the weights of the sub-models is called the weighted average ensemble. The weighted average is a technique, where we will have multiple sub-models which were trained on some train dataset and we average these models' weights to get single predictions.

This technique is called Model averaging. This technique can be improved by taking more weights by the model which is performing well on unseen data.

In simple terms, we take more contribution from the well-performing model on unseen data and less contribution from the model which is not performing well on unseen data. This method would significantly improve the accuracy of the model across test data.

A further approach to this is we can build a learning algorithm that can learn how to take the contributions of each trained sub-models. This technique is called Stacking Generalization or simply stacking [3] and can result in better prediction compared to single sub-model prediction.

This is something we train a completely new model to take best contributions from each sub-model. This method will reduce the variance of the model as we train the model to take contributions and thus increase the accuracy over the unseen data.

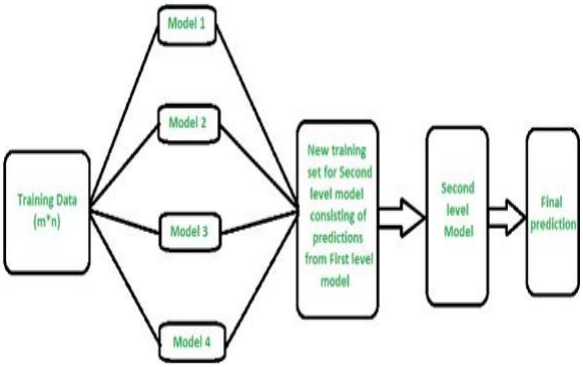


Fig 3. Stacking.

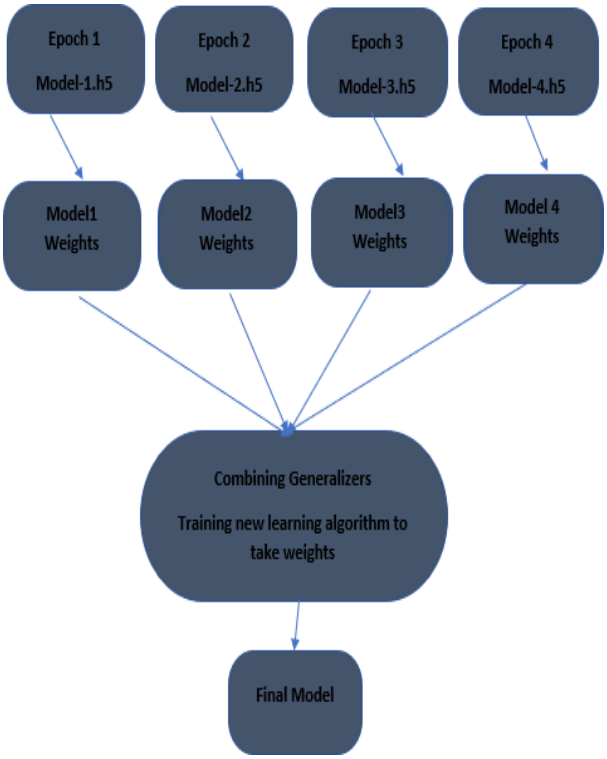


Fig 4. Stacking Generalization.

Let X be the independent variable and Y be the target variable or labels

Let D = {(X, Y)} be the training dataset given to the base-models.

Let B be the base models and M be the Meta models

Algorithm 1: Stacking Generalization
Input: Training dataset D = {(X,Y)}
Output: prediction – Stacked generalized prediction from the model

- **Step 1:** Train the base-level models
- for each t in Train
- Train a base model Bt ∈ B based on D
- end for
- **Step 2:** Train the meta-level models by
- for each t in Train
- Train a meta-model Mt ∈ M by giving the base models B as the input
- end for
- **Step 3:** Obtain the stacked ensemble prediction and return it.
- **Step 4:** return prediction.

Fig 5. Stacking Generalization Visualization.

IV. HORIZONTAL FOR CLASSIFICATION PROBLEMS

In Deep learning modeling problems, in most of the case, it is very tough to improve the performance of the model. Also, if we train for some number of epochs model often will have high variance.

It is very challenging to decide which model is performing well on unseen data at the end of training and we don't have clarity also which model will do well on all the cases.

Horizontal voting is one of the best ensemble methods to overcome this problem. Suppose if we train a model for 100 epochs, save the last 20 models. These last 20 models can be ensembled to get a single prediction. In simple terms, horizontal voting is a technique where we take votes from each sub-model and finalize the class which got the maximum votes.

This final ensemble prediction will do better compared to the randomly saved model. This method can be implemented when we have less data and can significantly improve the performance of the model. One key thing here is, we have to use an odd number of sub-models as we should not get an even number of votes for the class which will lead to a tie.

Let epoch be the number of iterations to train the model where e ∈ E. Let threshold be the threshold value greater than which the corresponding model will be saved.

Let model be the neural network models being trained on the training data Train (X, Y) such that m ∈ model.

Let 'save' be the function for saving the module in the location 'models/' directory.

Algorithm 2: Horizontal saving.
Input: threshold and epoch
Output: The models file with .h5 as an extension after the threshold epochs will be saved in the directory

- **Step 1:** Create a class that inherits the TensorFlow call back module
- **Step 2:** Save model after each epoch in the h5 format
- for each e in epoch
- if epoch > = threshold, then
- filename ← "model"+str(e)+".h5"
- save(filename)
- end for
- **Step 3:** end

Here we don't save the model after each epoch, because initially the model will have high variance and the loss will be more. As the model is trained for certain epochs, the loss function will go on decreases. So, after a certain threshold, the model will be saved. For example, if we are training the model for 500 epochs, saving all the models at each epoch will lead to 500 different models.

The models say the first 100-200 model performance will be less compared to the last 20 models. So, we can save the last 20 or 50 models so that these models will have less variance compared to early models and on ensemble, the model averaging or voting can give the best result compared to the random saved model.

Here in the example for 500 epochs, the threshold is 480. We are saving the models with epochs greater than or equal to 480 for 500 epochs. Ensemble of last 50 models will give good results compared to the randomly saved model.

Let model be the neural network models being trained on the training data Train (X, Y), such that m ∈ model

Let all_models be a list containing all the models. The location 'models/' directory.

Algorithm 3: Loading all the saved models

- **Input:** the models present in the location 'models/' to all_models
- **Output:** list of all the models
- **Step 1:** Initialize all_models as an empty list
- all_models = [ ]
- **Step 2:** Append all the models present in the location 'models/' to all_models
- **Step 3:** for each m in model, do
- all_models ← append(m)
- **Step 4:** return all_models

Let model be the set of neural network models being trained on the training data (X, Y), such that m ∈ model. Let yprediction be the predictions obtained by all the models on the test data Train (X', Y').

Let 'array' be the function for converting lists to arrays

Algorithm 4: Horizontal Voting for classification problem

- **Input:** models, test set Test' (X ', Y '), and empty y prediction list
- **Output:** predictions – final prediction obtained
- **Step 1:** Obtain the predictions of each model
- for each i in range(X)
- for each model in models, do
- y prediction [i] ← predict(X[i])
- Calculate highest number of votes for ith test data and append
- Y prediction [i] ← highest voted class
- end for
- end for
- **Step 2:** Convert list into an array
- Y prediction ← array(y prediction)
- **Step 3:** return y prediction

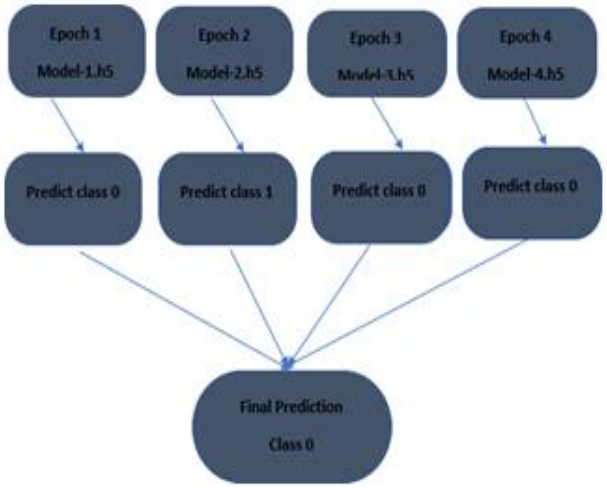


Fig 6. Horizontal voting for Classification.

Here is an example of a classification problem. In the above example we can see, we have saved the model after every epoch and we are loading the model. Each model will give its predictions; we can see 4 models with its prediction. 3 models are predicting with class 0 and 1 model prediction with class 1. With Horizontal Voting, the final prediction is class 0 as it has more votes.

V. HORIZONTAL SAVING AND MODEL AVERAGING FOR REGRESSION PROBLEMS

Model averaging is an ensemble technique where we take each model's weights for prediction. Each model will be equally contributed to the final prediction. In the regression problem, we will predict the continuous value.

Here model will be trained and saved as usually mentioned above based on certain epochs. We are not saving the model for all the epochs because, at the beginning epoch, the mean squared error and mean the absolute error will be high.

So, saving the model for each and taking it for prediction and averaging it, will have high error as initially, epochs models will have high mean squared and mean absolute error.

On averaging these models, the final prediction will be affected by these models, so we will save the last 50-100 epochs so that on averaging our prediction will be best compared to the randomly saved model.

Let model be the set of neural network models being trained on the training data Train (X, Y), such that m ∈ model. Let yprediction be the predictions obtained by all the models on the test data Test (X, Y).

Let 'array' be the function for converting lists to arrays

Algorithm 5: Horizontal Voting for a regression problem

- **Input:** models, test set Test (X , Y)
- **Output:** predictions – final prediction obtained
- **Step 1:** Load all the saved models and obtain the predictions of each model and append into the list yprediction
- for each model in models, do yprediction ← predict(X) yprediction ← append(yprediction)
- end for
- **Step 2:** Convert list into an array
- yprediction ← array(yprediction)
- **Step 3:** Find the sum of the arrays
- result ← sum(yprediction)
- **Step 4:** Calculate the average of each prediction and return the average
- average ← (result)/(total number of models)
- **Step 5:** return average

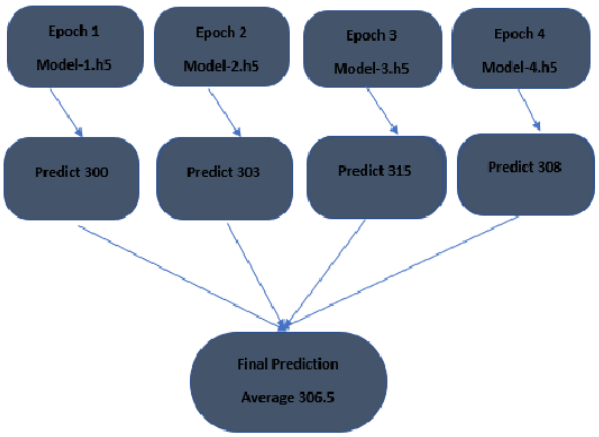


Fig 7. Horizontal voting for Regression.

VI. HORIZONTAL VOTING DEEP LEARNING ENSEMBLE WITH DIFFERENT MODELS

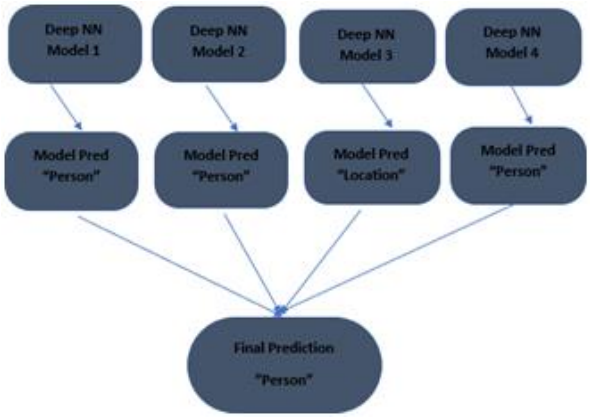


Fig 8. Horizontal voting with different models.

Horizontal voting with different models is the same as the above-explained ensemble technique. In the above technique, we have saved models after every epoch and took a final prediction with more votes. Here 90% of the concepts remain the same but here instead of saving the model with each epoch and loading to predict and taking votes, we train multiple Deep Neural Networks and combine the prediction of all the models and make a final prediction based on the maximum votes.

VII. EXPERIMENTAL STUDY

We have taken some common machine learning and deep learning datasets to test the ensemble learning algorithms. The result we found out was amazing. Without any hyper-parameters tuning etc; we could able to achieve good benchmark results.

For regression problems we tested the ensemble algorithms with the metrics mean squared error (mse), root mean squared error (rmse), mean absolute error (mae) and, r2 score. For classification problems, we tested the ensemble algorithms with the metrics accuracy, precision, recall, and f1 score.Some of the key observations made during the experimental study are:

- Neural networks are non-linear and have high variance.
- This can be frustrating while preparing the final model for predictions.
- Ensemble learning methods combine the model weights from different models to reduce the variance of predictions.
- Ensemble learning can also reduce generalization error.

Table 1. Horizontal Voting

Models	Precision	Recall	F1 Score	Accuracy
Random	0.68	0.61	0.64	0.83
Best 1	0.78	50.59	0.67	0.84
Best 2	0.76	0.60	0.67	0.85
Ensemble	0.78	0.61	0.68	0.87

Table 2. horizontal Voting (Averaging)

Models	MSE	RMSE	MAE	R2 Score
Random	2989449448.8	54675.8580	35326.6222	0.76629
Best	2889449448.8	53753.599	33326.6222	0.76.21
Ensemble	2748419021.2	52425.3662	322236.66	0.78514

Table (1) shows the experimental result on the classification dataset. The random model is the model which was saved using the model. Save () method using TensorFlow. The random model is not the best. Best 1 model is the model which was saved using callbacks. This model is the best model through the epochs on which the model was trained, which means it has the low validation loss and highest validation accuracy.

The best 2 model is another model with different neural network architecture and this was saved using callbacks with low validation loss and highest validation accuracy. Here all 3 models came out with accuracy score around 83% to 85%.

On using the horizontal voting ensemble method for these 3 models, the accuracy of the model was jumped to 87 %. Although a 2% increase looks very low in reality, it is a

very good improvement when we deal with large datasets. Also, to make a note, both recall and f1 scores improved.

Table (2) shows the experimental result on the regression dataset. Both random and best models were saved using the same neural network architecture. During the last 10 to 20 epochs, the model was saved at every epoch. And these models were used for horizontal averaging. The final model after averaging these 10 to 20 models had a low mean squared error and high R2 score which was found out to be 2748419021.1 and 0.78 respectively.

Table 3. Stacking Generalization

Models	Accuracy
Model 1	0.817
Model 2	0.819
Stacked Model	0.83

Table (3) shows the experimental result on a classification dataset. Model 1 and Model 2 were the two different neural network architectures that were trained with an accuracy of 81.7% and 81.9 % respectively.

Using the Stacking generalization ensemble method these two models were stacked and the accuracy of this model was jumped to 83 %. There are many ways in improving the performance of neural networks. Ensemble technique is one of them.Here are the snapshots of few different Neural Networks architecture used for ensemble.

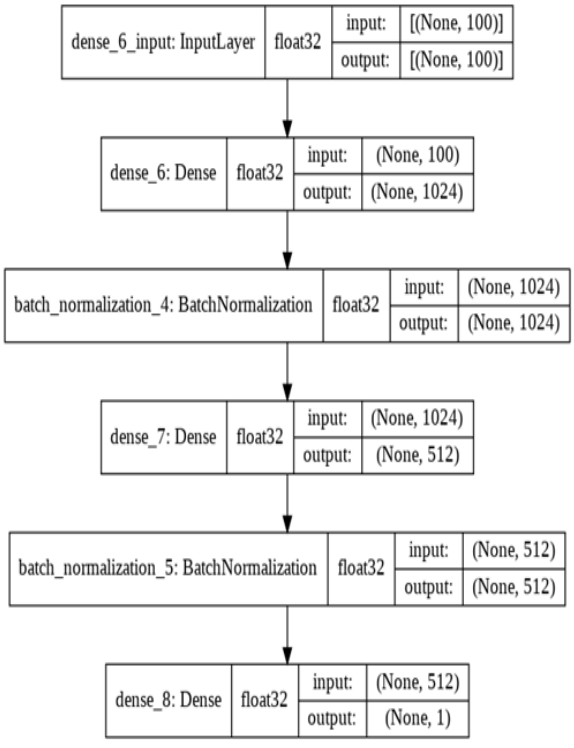


Fig 9. Neural Architecture 1.

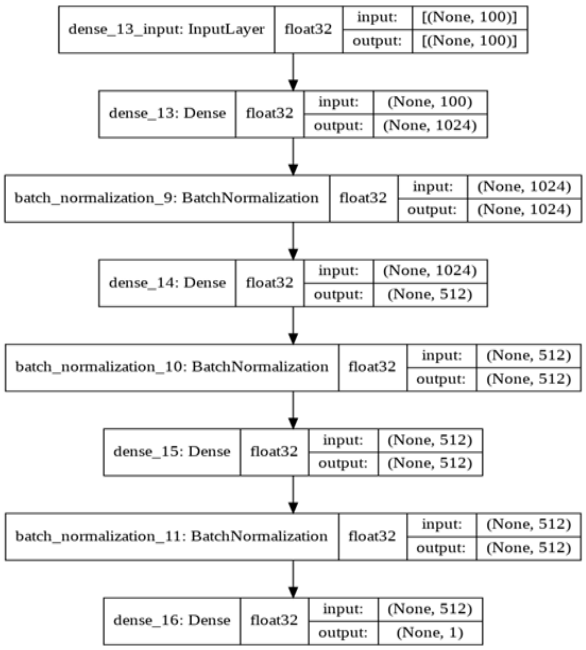


Fig 10. Neural Architecture 2.

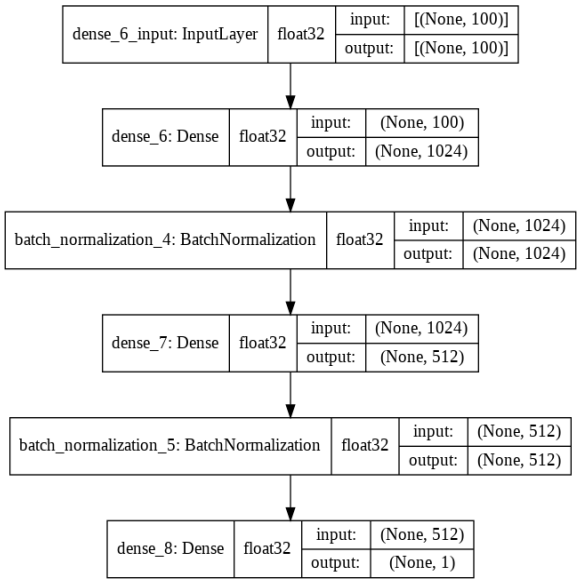


Fig 11. Neural Architecture 3.

VIII. CONCLUSION

Neural Networks is a concept where it exactly replicates our human brain's neural network system. It is often very difficult to build State of the Art (SOTA) models. We have so many different methods to improve the performance [4] [5].

This might take time to come to a conclusion which method would work out for the particular problem. But for ensemble suppose if we have 2 to 3 models performing decently on the unseen dataset, the ensemble method can improve the performance of the model on an unseen dataset.

And if we don't have 2 to 3 models, we can save the model at each epoch and ensemble it. In this, we should different methods of ensemble techniques that can improve the performance, yet there is still research going on and better algorithms can be applied and build a strong generalizer.

REFERENCES

[1] L. K. Hansen and P. Salamon, "Neural network ensembles," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993-1001, Oct. 1990, doi: 10.1109/34.58871.

[2] Y. Liu, X. Yao, Ensemble learning via negative correlation, Neural Networks, Volume 12, Issue 10,1999, Pages 1399-1404, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(99)00073-8.

[3] David H. Wolpert, Stacked generalization, Neural Networks, Volume 5, Issue 2, 1992, Pages 241-259, ISSN0893-6080, https://doi.org/10.1016/S0893-6080(05)80023-1.

[4] MacKay D.J.C. (1995) Developments in Probabilistic Modelling with Neural Networks — Ensemble Learning. In: Kappen B., Gielen S. (eds) Neural Networks: Artificial Intelligence and Industrial Applications. Springer, London. https://doi.org/ 10.1007/978-1-4471-3087-1_37

[5] Polikar R. (2012) Ensemble Learning. In: Zhang C., Ma Y. (eds) Ensemble Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978- 1-4419-9326-7_1

INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078

(Volume 7, Issue 3 - V7I3-2048)
Available online at: https://www.ijariit.com

Masking private user information using Natural Language Processing

Satwik Ram Kodandaram
satwikram29@gmail.com
Visvesvaraya Technological University, Bangalore, Karnataka

Kushal Honnappa
kushal.h1999@gmail.com
Presidency University, Bangalore, Karnataka

Kunal Soni
sonee069@gmail.com
Sanskar School, Jaipur, Rajasthan

ABSTRACT

The Internet brings a lot of efficiency to our lives but we must be aware that everyone exchanges a huge amount of data while interacting with the internet. One of the most important leisure activities one gets from the internet is to be able to socialize. For example, social media has become part and the core of our lives. With more than 2.3 billion active users, data privacy is an issue of concern. The world of the internet has become full of frauds hunting for personal information they leverage for their immoral activities. So, coming up with an algorithm that could secure data and process it such that no private data is involved, and machines continue to be trained with greater data. This could mean a dataset with data that is processed in a manner to make it anonymous. If there is any private information, we will mask that information with pseudo data. We use Named Entity Recognition using Deep Learning for identifying and masking personal information. In this paper, we will discuss how we mask private data. This model will be a successful technique to hide one's personal information to achieve complete data privacy.

Keywords: *Named Entity Recognition, Deep Learning, Social Media Platforms, Data privacy; Masking, Anonymity*

1. INTRODUCTION

Social media user's concerns about their data privacy have spiked in recent years. Incidents of data breaches have alarmed many users to rethink their relationships to social media and the security of their personal information. The dramatic story of consulting agency Cambridge Analytica is an example that exploited the private information of over 50 million Facebook users to influence the 2016 American presidential election. These issues are not acceptable with private user information. According to the study conducted by the Pew trust, 80 percent of social media user's information being concerned about businesses and advertisers accessing and using their social media posts. These privacy issues have prompted the advocacy of tighter regulations.

People will hesitate to share on social media as their data can be used or leaked. Given today's social media privacy issues and concern, skilled cybersecurity professionals will play a vital role in protecting the private user information and also the application needs some automation which can hide some private data.

Fig. 1. An overview of privacy issues concerning the type of social media data Tasks

Fig. 2. Where are People concerned about Online Privacy

1.1 Motivation

Data privacy has become a very important topic in our lives as we are moving towards a more digitalized lifestyle. Crucial data is being collected of people when they are in different mental states through various digital devices installed in their homes or workplace. This data is later used to predict their behavior and maybe show advertisements accordingly. Even to train our machines for machine learning, we need to give them data, the data which is exposed to a larger group of people. But we are trying to anonymize this, anonymize the collected data, so even if it lands in the wrong hands, it holds no private information. This data can then be used for various other purposes as it still contains the gist of the whole idea.

1.2 Problem Statement

In the present scenario, we see a lot of people are facing mental health issues. This is mainly because of the stress which they face. This happens mainly due to the changed lifestyle of people. So mental health has become a need for all of us that needs to be addressed. Being physically fit is important, but at the same time, being mentally fit is also equally important for all of us. We have a lot of social media applications where users cannot share their thoughts/views.

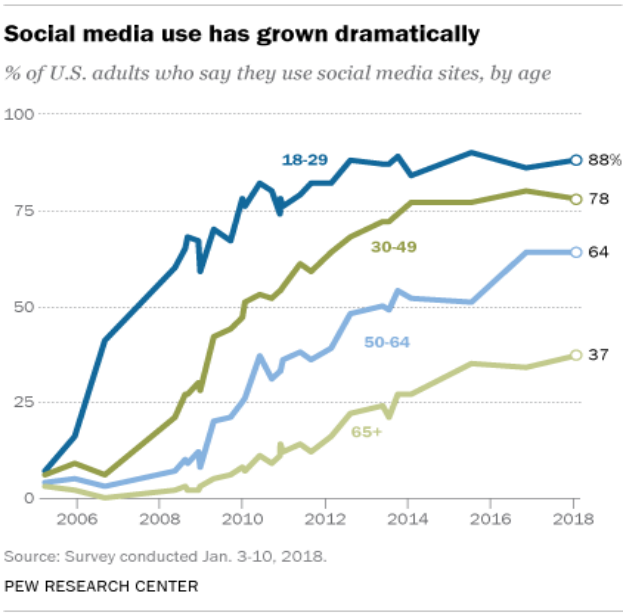


Fig. 3. Social Media usage growth

Below listed are some of the issues, why users cannot express their thoughts/views:

- ✓ Most of the social media application is collecting private information which can be hacked in the future and that information could be leaked to hackers who can misuse the data
- ✓ There is no complete user anonymity: Users will have to share their email with the website and then are allowed to post anonymously, which is not complete anonymity.
- ✓ Less accessibility: Most of these platforms are limited to the English language only. Moreover, the support for people who can't type is missing at most sites, which makes these platforms less accessible to the public.
- ✓ Targeted advertisements: Some platforms show targeted advertisements based on what users share. This could be a helpful feature, but this is something that is not beneficial when dealing with mental health. Users feel they are being tracked.
- ✓ Disclosure of private information on social media applications: Although users are allowed to post anonymously, sometimes, the data they share contains some private information about other people. This can lead to unexpected problems.

One of the solutions for the above-mentioned problems is an application where there is complete user anonymity. Complete user anonymity can be achieved by masking user's private information with pseudo data. We train Deep Learning algorithms to identify these private user's information and then mask this private information with pseudo data. We use Named Entity Recognition for identifying entities like Name, place, etc, and then we will mask it. By this, we can able to hide private user's information and achieve complete user anonymity.

1.3 Objectives

Our objective is to solve the problem of data privacy. We plan on removing the sensitive part from a given piece of data to ensure that no private information like names and addresses are present. All this data can be masked with pseudo data of the same category like mobile numbers, names, and addresses. This will keep the data legitimate as a whole and will also remove all the sensitive information it collects. This could be one of the methods to achieve complete data anonymity.

2. LITERATURE REVIEW

Currently, there are a lot of works that de-identify sensitive data and mask it with different approaches. This can be extended to different languages given a large dataset is available for the data:

2.1 Towards Personal Data Identification and Anonymization Using Machine Learning Techniques [1]

In this paper [1], they have implemented using Supervised Machine Learning Algorithms. If we use Deep Learning algorithms we can fine tune it and extend our model for another use cases using Transfer Learning concept but not possible using Machine Learning algorithms.

2.2 De-identification in Natural Language Processing [2]

This paper [2] focuses on the usage of NLP for de-identification and the importance it in different areas like medical, social media, and CVs and describes what data need to be preserved and removed.

2.3 An Introduction to NLP-based Textual Anonymisation [3]

This work by Ben Medlock [3] talks about building a corpus and the process of construction of the same. He critically evaluates the system and talks about the issues he faced working on the project. He also introduces the HMM-based tagger which could be used as a corpus for the anonymous data.

3. IMPLEMENTATION

3.1 Deep Learning

Deep Learning is a subfield of Artificial Intelligence where it replicates the human brain, human Neural System. Deep Learning can be supervised, semi-supervised or unsupervised. Deep Learning is capable of learning unsupervised from data that is unstructured or unlabelled. Deep Learning algorithms enable us to train machines and make machines to understand the data and make decisions based on the correlation that exists between the dataset, the same way how humans think to make decisions with billion neurons connections. Deep Learning models are very slow to train and require high computational power, nowadays GPU or TPU have become a requirement to execute the deep learning algorithms.

Although GPUs are very expensive yet without them training deep neural networks to high performance is practically not feasible. Today Deep Learning models are achieving the State of the Art (SOTA) on challenging machine learning problems like language translation from one language to another.

3.2 Natural Language Processing

A subfield of Artificial Intelligence is natural language processing. NLP's main goal is to understand and react to human languages. Like other Machine learning algorithms, NLP requires data to be trained. Working with text is very important and it is very hard as it requires knowledge from a diverse domain such as Linguistics, Statistics, Machine Learning, and these days Deep Learning. When data are trained with NLP models this algorithm tries to learn the language on its own with the help of available data its learning process is similar to humans learning natural languages. The advancement in NLP technology helps the world to grow better and faster. NLP helps humans in many aspects such as translation from one language to another this reduces the human's burden in knowing all the languages. NLP help's us to identify the tag of a given word, for example, "Bengaluru" with "geo-loc" as a tag.

The chatbot is an application of NLP that can be used in Customer support, Schedule a Meeting, Product Suggestions, Order Pizza, and so on. Usage of chatbots in these areas not only reduces workload but also saves customers waiting time. NLP has the skill that reads, writes, and also speaks the given languages the same as humans do. Combining all these techniques an application can be built that can behave, interact the same as humans without giving an impression of a machine.

3.3 Named Entity Recognition

Named Entity Recognition (NER) also known as Named Entity Identification, entity chunking, and entity extraction is a subtask of extraction of information from a corpus of sentences. It helps us to identify named entities like a person, location, event, organization, etc which are already pre-defined.

Fig. 4. Named Entity Recognition

Extracting main entities like a person, location, etc. helps us to sort unstructured data and detect important information, which is crucial if we have to deal with a large corpus of datasets. Named Entity Recognition can be achieved using Deep Learning classification where we give a set of training examples with labels stating words with the corresponding entity as labels/class to the model. On training Deep Neural Networks model with probability function can able to predict the class which word belongs to. There are a lot of applications on Named Entity Recognition, in our application, we use mainly named entity recognition to identify tags like a person, location, etc, and mask this information to hide the private information of the users and make our application complete user anonymous.

```
<xdrs xml:id="1">
 <taggedtokens>
  <tagtoken xml:id="il001">
   <tags>
     <tag type="tok">Thousands</tag>
     <tag type="pos">NNS</tag>
     <tag type="numex">O</tag>
     <tag type="animacy">Human</tag>
     <tag type="timex">O</tag>
     <tag type="lemma">thousand</tag>
     <tag type="to">9</tag>
     <tag type="senseid">thousand.n.01</tag>
     <tag type="sense">1</tag>
     <tag type="from">0</tag>
     <tag type="namex">O</tag>
   </tags>
  </tagtoken>
  <tagtoken xml:id="il002">
   <tags>
     <tag type="tok">of</tag>
     <tag type="pos">IN</tag>
     <tag type="numex">O</tag>
     <tag type="animacy">O</tag>
     <tag type="timex">O</tag>
     <tag type="lemma">of</tag>
     <tag type="to">12</tag>
     <tag type="sense">0</tag>
     <tag type="from">10</tag>
     <tag type="namex">O</tag>
   </tags>
  </tagtoken>
  <tagtoken xml:id="il003">
```

Fig. 5. Data in XML

363436	joining	O
363437	Spaniard	B-gpe
363438	Rafael	B-per
363439	Nadal	I-per
363440		O
363441	Swiss	B-gpe
363442	Roger	B-per
363443	Federer	I-per
363444		O
363445	Serbian	B-gpe
363446	Novak	B-per
363447	Djokovic	I-per
363448		O
363449	Scott	B-per
363450	Andy	I-per
363451	Murray	I-per
363452	and	O
363453	Swede	B-per
363454	Robin	I-per
363455	Soderling	I-per

Fig. 6. XML to CSV

3.4 Data Collection and Data Pre-processing

3.4.1 Dataset: We have collected data from The Groningen Meaning Bank (GMB) which has a corpus of English texts with deep semantic annotations. The dataset has more than 1.4 million different tags, each tag representing different named entities with their corresponding words. GMB is an adequately large corpus with a lot of annotations. Unfortunately, GMB is not perfect. It is not a gold standard corpus, meaning that it's not completely human-annotated and it's not considered 100% correct. The corpus is created by using already existed annotators and then corrected by humans where needed. Here are the following classes in the dataset -

- ✓ geo = Geographical Entity
- ✓ org = Organization
- ✓ per = Person
- ✓ gpe = Geopolitical Entity
- ✓ tim = Time indicator
- ✓ art = Artifact
- ✓ eve = Event
- ✓ nat = Natural Phenomenon

The attached dataset is in tab-separated format; the goal is to create a good model to classify the Tag column. The dataset is labeled using the IOB tagging system.

The dataset is in XML form. The XML tree has words with all types mentioned.

We have converted XML form to a structured Data Frame, that is to CSV format. We have parsed over the XML tree, extracted the words and their corresponding tags, and converted them to CSV files.

3.4.2 XML to CSV Conversion: Let XML file be the Input that needs to be converted to CSV file. We will pass two empty lists that are words and tags list which need to be parsed and appended.

Let result be a list that consists of all the punctuations and other symbols which need to be filtered out from the XML file.

Algorithm 1: XML to CSV

Input: XML file, words list, tags list, and result
Output: Converted Xml to CSV

1. **Step 1:** Parse the XLM tree
2. tree ← **parse**(file)
3. **Step 2:** Get root element from tree
4. root ← tree.**getroot()**
5. **Step 3**: Parsing throgh root and fetching words and tags
6. **for each** elem **in** root, **do**

for each subelem **in** elem.**findall**(tags), **do**
if subelem attribute type == 'tok'
if subelem text in result
do nothing, **pass** or **append(NaN)**
else
words ← **append**(text)
elif subelem attribute type == 'namex'
tags ← **append**(text)
end if

7. **end for**
8. **end for**
9. **Step 4: Create Data Frame from list**
10. df = DataFrame({"words": words, "tags":tags})
11. **Step 5: Convert Data Frame to CSV**
12. df.**to_csv**('ner.csv', index = 'false')

3.5 Tokenization

The process of splitting a sentence or a phrase into smaller units that will be individual words, this split piece of words is called as tokens. For example, let's take a sentence "Alex knows to program." on tokenizing we get ["Alex", "knows", "to", "program", "."] this tokenization [4] [5] is done by word boundaries. With the help of this tokenizer, we will be able to count the number of words in the text and count the frequently occurring words. After tokenization is done, we encode the tokens to numeric format.

3.6 Word Embeddings

Word Embeddings are created using neural networks with one input layer, one hidden layer and, one output layer. Neural Network doesn't understand the raw text given as input. We should encode it to the numbers using one-hot encoding or tokenization. Word Embeddings are like a numerical representation of a text. It is a type of word representation and learned representation that has words with similar meanings to have similar representation. Word Embeddings is a technique where individual words are represented as real-valued vectors which are predefined in the vector space of corpus. Each word is mapped to one vector and the vector values are learned in a way that resembles a neural network, which is more efficient while predicting.

Example of Predicting the next word:
Student Opened their _____
The corpus has houses, books, lamps, and stamps.

Here the prediction should be: Student Opened their books.

Numerical Representation of Words houses books lamps stamps <0.6, 0. 2, 0.1, 0.1>

How it is represented in Word Embeddings Each row of W contains feature weights for the corresponding word in the vocabulary.

$$W = \begin{pmatrix} 1.2 & -0.3 & 0.9 \\ 0.2 & 0.4 & -2.2 \\ 8.9 & -1.9 & 6.5 \\ 4.5 & 2.2 & -0.1 \end{pmatrix} \begin{matrix} \text{books} \\ \text{houses} \\ \text{lamps} \\ \text{stamps} \end{matrix}$$

X will be the input given to the model to predict.

Each dimension of X corresponds to a feature to the prefix

$$X = < -2.3, 0.9, 5.4 >$$

We do Matrix Multiplication of W weights and input X to obtain W*X

$$W*X = < 1.8, -11.9, 12.9, -8.9 >$$

On top of W*X, a probability function is applied say softmax function for multi labels.
Softmax Function:

$$\text{Softmax}(X) = \frac{e^x}{\sum_i^n e^i} \qquad (1)$$

$$\text{Softmax}(W*X) = < 0.24, 0,73, 0.006, 0.002 >$$

The highest Probability is 0.73 that is for books so the prediction books.

Student Opened their books.

If the prediction is wrong, the Word Embeddings weights are adjusted to predict the right word.

3.7 Recurrent Neural Networks

We, humans, understand a sentence by understanding each word from it. A word is meaning changes concerning its past work. Similarly, neural network needs past event's information to understand now happening or future accruing events. This cannot be achieved by a conventional neural network.

This cannot be achieved by a conventional neural network. RNN [6] does this work, it does it by a network which is looped, this caries flow in information. An RNN is a duplicate of the same neural network where each one of it passes information to the other.

RNN works well when it needs information from a previous word from the same sentence. But it fails when it needs information from the previous sentence for example "Bangalore is capital of Karnataka."

In this sentence, Karnataka is predicted with the help of previous information. But in the sentence "Raju is a fisherman, so he catches fish daily" to predict the work fish there is a huge gap with the word fisherman. Here RNN fails to handle these "long-term dependencies". So, in these cases we use LSTM.

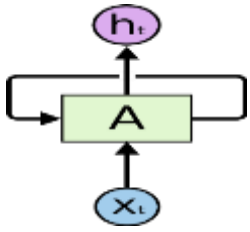


Fig. 7. RNN with its loops

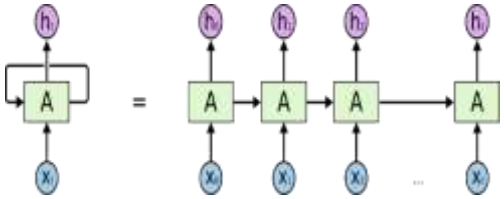


Fig. 8. RNN which shows that it is a combination of the same neural network

3.7.1 Long Short-term Memories (LSTM): LSTM [7] is a kind of RNN [6] that works well for long-term dependencies and also can remember the information for a longer period. It is capable of processing an entire sequence of data.

3.7.2 Difference between RNN and LSTM

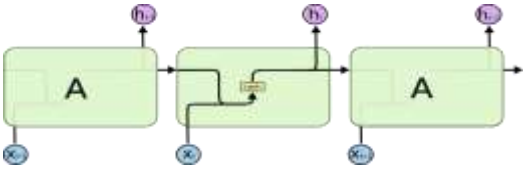


Fig. 9. RNN

In traditional RNN fig 9, we have only one tanh layer but in LSTM we have 4 interactions as shown in fig 10.

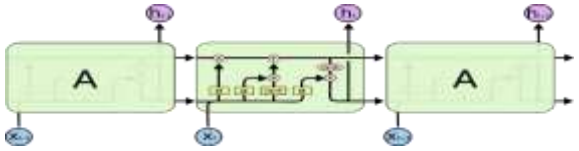


Fig. 10. LSTM

In fig 10 we can see there are many notations involved. Let's understand it by looking into fig 11.

Each line in fig 11 carries an entire vector from one node's output to the inputs of others. The learned neural network layers are represented by yellow boxes. The pink circle represents pointwise operations such as vector addition.

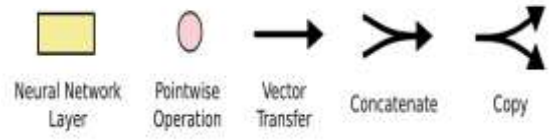


Fig. 11. Working LSTM

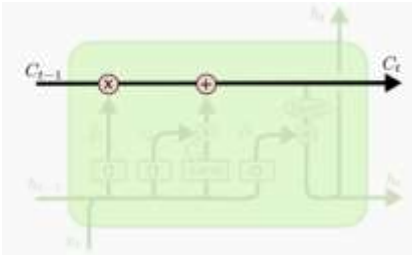


Fig. 12. Cell State

The alternation in the information like removing or adding the information is done using a structure called gates fig 13. These gates are made up of sigmoid neural network layers and pointwise multiplication operations.

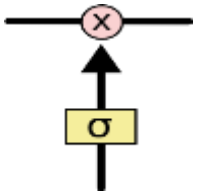


Fig. 13. Gates

This sigmoid layer gives output as 0 or 1. When it's 0 none of the information is left passed. If it's 1 then all the information is passed through it.

This works fine when we want to predict the future word or future event. But how can we predict the middle word for example fill in the blanks question? In this case, we need knowledge of the previous word and also the next word of the blank space. This cannot be solved using LSTM. Here comes the concept of Bidirectional LSTM. Let's see that is a Bidirectional LSTM (Bi-LSTM).

3.7.3 BI-LSTM (Bi-Directional Long Short-Term Memory)

Bidirectional LSTM [8] is a sequential model which consists of two LSTM [7]. Where one process in the forward direction and another process in the backward direction. This model not only helps to predict immediately following words but also precede word.

This model works well in our implementation as in our project we need to recognize the tags such 'O' tags this includes as

shown in fig 2, person name, location name. To detect these tags the model needs information on both adjacent words for example in the sentence "Ram went to Mumbai" here to predict Ram is a name of a person it needs knowledge of the future word. Similarly, to predict Mumbai as a location name it needs knowledge of precede word.

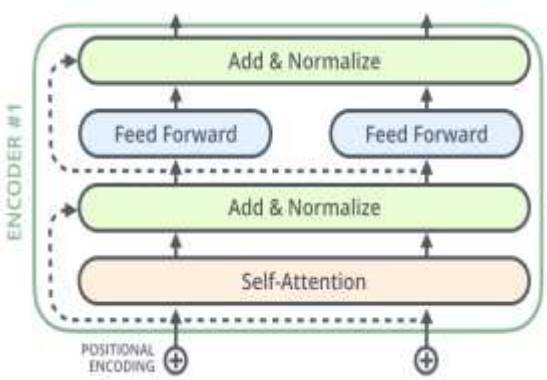


Fig. 14 Transformer

3.7.4 Transformers and Multi-head Self-Attention Mechanism

The Transformer was proposed by Google AI Team in the research paper [9]. The Transformer outperforms Google's Machine Translation model in specific tasks. The biggest benefit of using a Transformer is how Transformer lends itself to parallelization. The Transformer a model that uses a self-attention mechanism to boost the speed at which neural network models can be trained. Self-attention can completely replace recurrence and helps to focus on particular words and their relationship with other words.

A transformer has several Encoders in it. The number of Encoders is a Hyper-Parameter. In the official Paper [9] Transformer has 6 Encoders. In Encoder we have, Positional Encoding, Self-Attention Layer, and Feed Forward Network with Residual Connections as shown in fig 10. On top of the Word Embeddings, we apply 3 different projections of linear layers vector space and obtain query, keys, and values. For every input word, we will apply linear layers and get Query, Keys, and Values. These Query, Keys, and Values come from the same text.

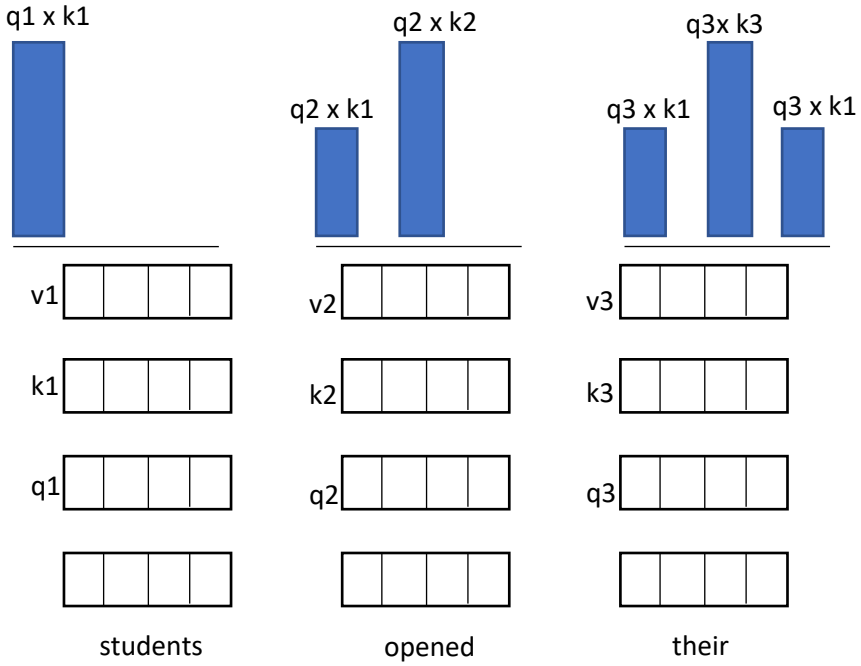


Fig. 15. Self Attention

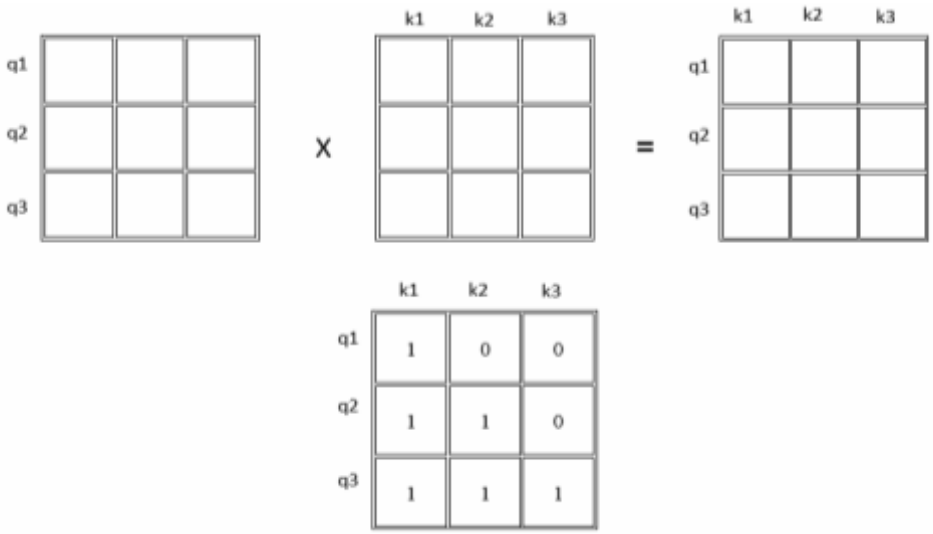


Fig. 16. Self Attention, Matrix Multiplication, and Masking

In the next step, we take the dot product of query and keys and obtain attention scores. The query and keys are used to compute the attention and values are used to compute attention-weighted representation.

For example, in predicting the next words, we obtain query, keys, and values for each word. On top of that, we obtain the probability. For the third word, we have distributed over all 3 keys. For the second word two distribution because here we don't include the third key in our attention weighted average as it is completely independent of the third time stamp and for the first word only one as mentioned in fig 16. Here in the matrix multiplication, we have masked with 0 but in reality, we will mask with some negative numbers.

We obtain z1, z2 and z3 as follows:

$$z1 = q1*k1 + v1 \quad (2)$$

$$z2 = q2*k1 + v1 + q2*k2 + v2 \quad (3)$$

$$z3 = q3*k1+v1+q3*k2+v2+q3*k3+v3 \quad (4)$$

This is how we get token-level representation. On top of these z1, z2 and z3 we apply SoftMax layer.

Here there is no dependency between zn and zn-1

Doing Parallel all attention computation by just matrix multiplication.

We will mask, after masking we apply SoftMax and we get valid attention distribution.

The same in the Transformer model we extend this to multiple heads. That means, we have many different projection matrixes and compute the attention. This operation is called a multi-head self-attention mechanism.

Many variants of attention are:

$$\text{Original } a(q, k) = w2T * \tanh(w1[q;k]) \quad (5)$$

$$\text{Bilinear Product: } a(q, k) = qTWk \quad (6)$$

$$\text{Dot Product: } a(q, k) = qTk \quad (7)$$

$$\text{Scaled dot Product: } a(q, k) = \frac{q^T k}{\sqrt{|k|}} \quad (8)$$

3.7.5 Positional Encoding: Attention models don't contain any recurrence or convolution, positional encoding is added to the model to give some information about the relative position of the words in a sentence.

This positional encoding is added to the embedding layer. Embedding represents a token in d-dimension space where tokens with similar meanings will be closer to each other in space.

But the Word embeddings don't give any relative positions of the words in a sentence. So, after adding positional encoding to the word embeddings, similar words will be closer to each other in the d-dimension space.

The formula to calculate the Positional Encoding is:

$$\mathbf{P_{i,j}} = \mathbf{sin}\left(\frac{\mathbf{i}}{\mathbf{10000}^{\frac{\mathbf{j}}{\mathbf{d_{emb-dim}}}}}\right) \textbf{ if j is even} \quad (9)$$

$$\mathbf{P_{i,j}} = \mathbf{cos}\left(\frac{\mathbf{i}}{\mathbf{10000}^{\frac{\mathbf{j-1}}{\mathbf{d_{emb-dim}}}}}\right) \textbf{ if j is odd} \quad (10)$$

3.7.6 Bert: BERT [10] is a trained Transformer Encoder stack. BERT has two variants, BERT Base with 12 Encoders and BERT Large with 16 Encoders. Transformer Encoders are the basic building blocks for BERT. The base for BERT is Transformer. Now we have open-source Pre-trained BERT available online, where we can change the last layer of the BERT and our custom function to perform our tasks.

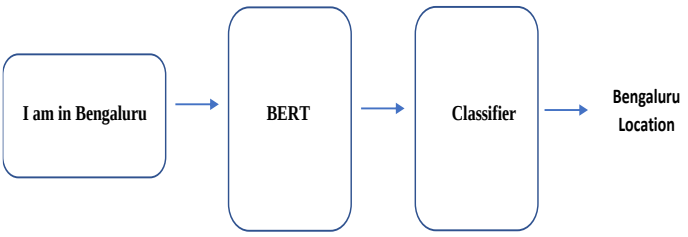


Fig. 17. BERT for Named Entity Recognition

Here we add our custom named entity recognition classifier on top of the BERT with SoftMax Layer using the transfer learning concept. We will include the top layers of the pre-trained BERT and we will train our model which will identify the named entities.

3.8 Proposed Model

We have collected the data from The Groningen Meaning Bank (GMB) which has a corpus of English texts with deep semantic annotations. It has around 1.4 million named entities. The data was in XML form, which we have converted from XML to CSV. The data has around 12 tags but we have taken only the essential tags that are required for our application.

After Data Pre-processing, we have tokenized the text and splitted the data into train and test. We are building different models using RNN, LSTM, Attention model, and BERT architecture and we will ensemble the predictions using Horizontal Voting to get a more generalized model.

This final model is used to predict the named entity like Name, location, etc. Once we identify these entities, we will mask this to some pseudo data.

Working Example:
Raw Text Input: "I am Satwik, I am in Bengaluru."
Output from the model: "I am XXX, I am in YYY."
This model is being deployed in Django Rest Frameworks with Flutter as frontend. By this user can post any data as data will be completely masked.

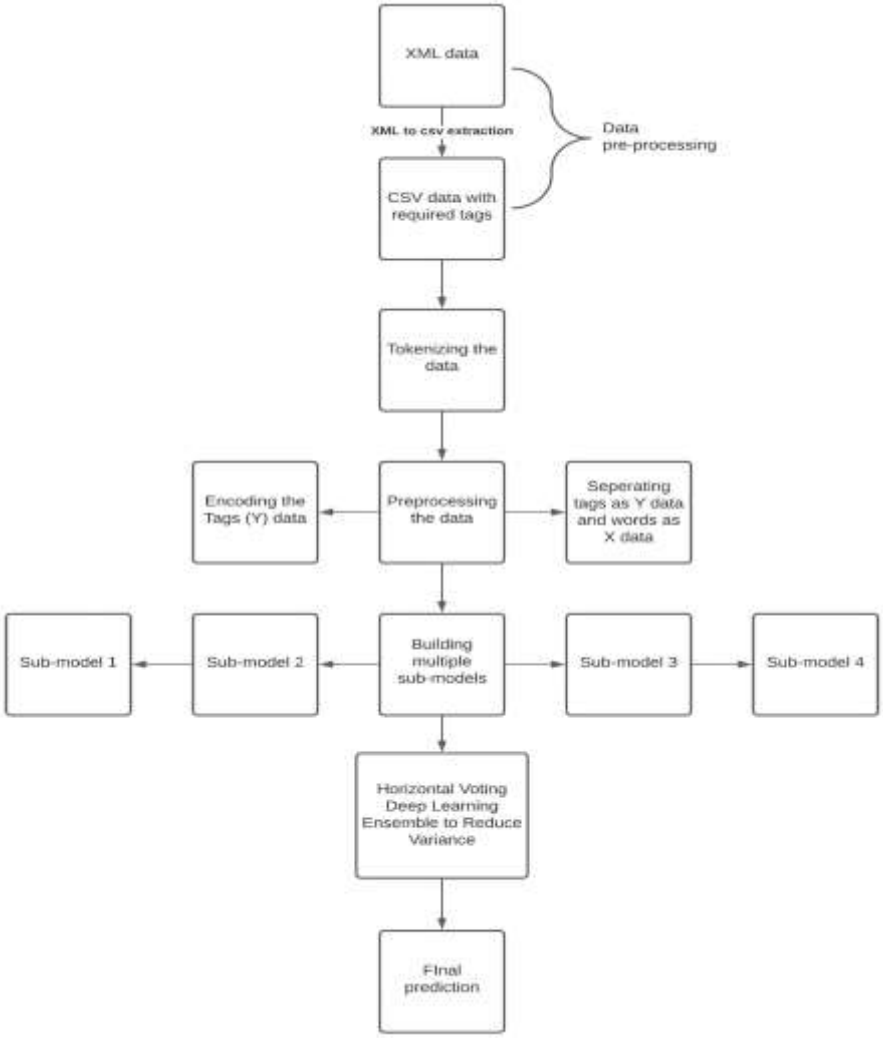


Fig. 18. Proposed Model

3.8.1 Horizontal Voting Deep Learning Ensemble to Reduce Variance with different models

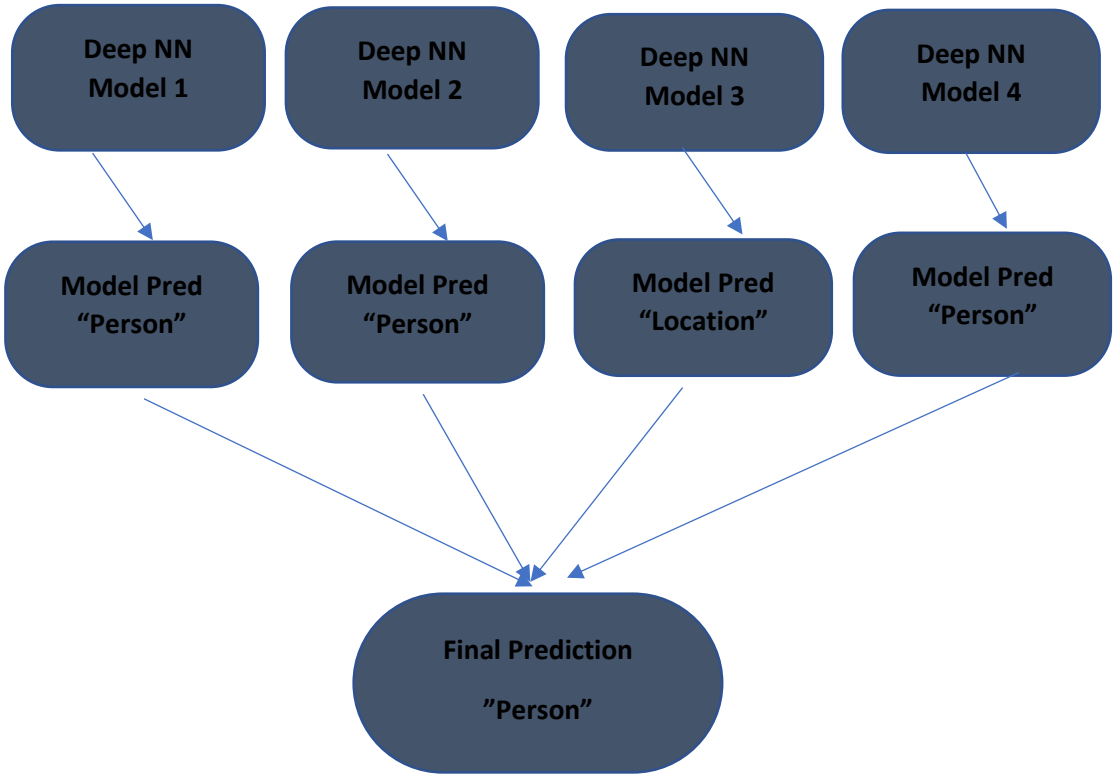


Fig. 19. Horizontal Voting Deep Learning Ensemble

Predictive modeling problems where we have a smaller number of training data relative to the number of unlabelled examples are challenging. Neural networks are used to train these kinds of problems and neural networks perform well on these types of the problem although they can suffer from high variance in model performance as measured on validation set or hold-out set.

This makes choosing the final model at end of the epoch is risky as there is no clear signal of which model is performing well compared to others towards the end of the training run.

As we have ensemble technique in Machine Learning [11] similarly we have Neural Networks Ensemble [12] [13] [14]. We have many ensemble techniques like Staking Generalization [15].

One of method in ensemble technique is the horizontal voting ensemble which is a simple approach to address this issue, where we have a collection of models saved, and these saved models are used as an ensemble that results in more stable and better performance on average compared to randomly choosing a single final model. This approach was developed specifically for those predictive modeling problems where the training dataset is small compared to the number of predictions to predict by the model.

In the above figure, we can see we have 4 Deep Neural Networks sub models each can have the same architecture or different architecture. Each sub-model predicts the probability of the class, here we can see prediction as "Person".

Here 3 sub-models predict as a person and 1 sub-model predicts as a location. So, the final model prediction is the "Person" class as it has 3 votes. Here, for example, we have taken 4 sub-models but in reality, we go with odd numbers of sub-models to avoid equal class prediction by sub-models.

Let model be the set of neural network models being trained on the training set T(xi,yi), such that m ∈ model. Let yhat be the predictions obtained by all the models on the test set T'(xi',yi'). Let 'array' be the function for converting lists to an array

Algorithm 2: Ensemble Prediction

Input: models, test set T'(x_i',y_i'), and empty yhat list

Output: predictions – final prediction obtained

1. **Step 1:** Obtain the predictions of each model
2. **for** each i in range(x_i)
3. **for each** m **in** model, **do**
   yhat[i] ← **predict**(x[i])
   Calculate highest number of votes for ith test data and append
   yhat[i] ← highest voted class
   end for
4. **end for**
5. **Step 2 :** Convert list into an array
6. yhat ← **array**(yhat)
7. **Step 3: return** yhat

4. EXPIREMENTAL RESULTS

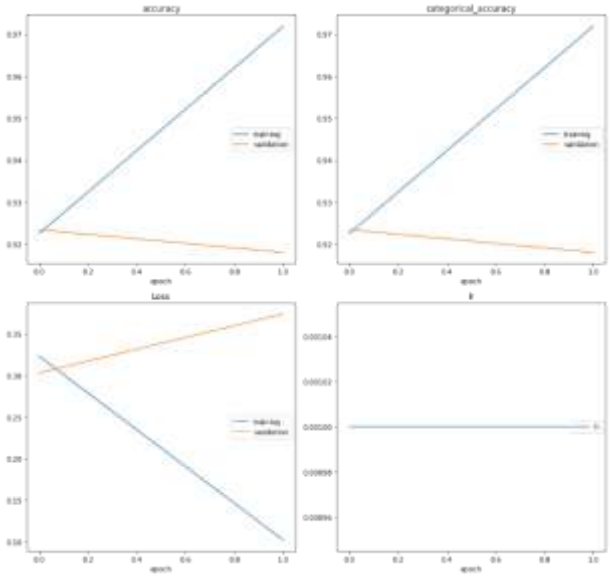


Fig. 20. Model Training plots

We have trained all the models for around 20-50 epochs with a batch size of 32.

We have done several experimented with our project by training our models with different algorithms, and with different approaches, we used BI-LSTM, Transformers, Bert. Here it was found that each model achieved good results.

For all the models while training we have used ReduceLROnPlateau callback with factor = 0.2, patience = 5, min_lr = 0.001 and Learning rate scheduler.

When a user enters a sentence like, "Ramesh lives in Bangalore." then the following result should be obtained "Ramesh" as name-tag, "lives" and "in" as o-tag, and "Bangalore" should be identified as loc-tag.

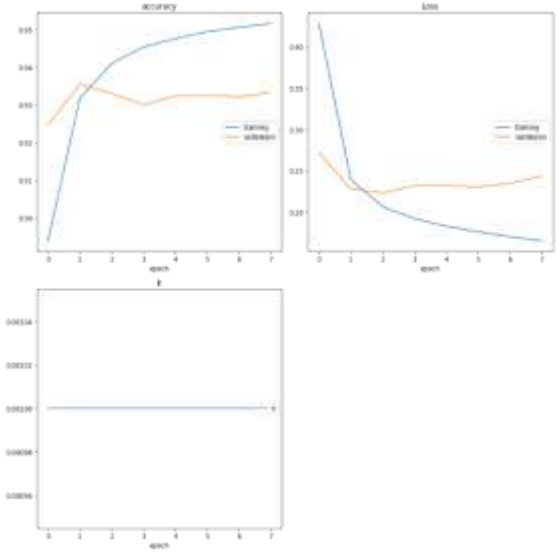


Fig. 21. Model Training plots

After achieving this we will be masking the information into the generalized form to provide anonymity for the users. For this example, we will be replacing name-tag and loc-tag with generalized form and only o-tags values will remain the same. The overall result will be formed as <person name> lives in <location name>. This can also be formed by creating a random character and replacing it instead of private data like "gtans lives in ytend" and explained as "gtans" is a person name and "yten" a location name. Here gtans and yten are the pseudo data.

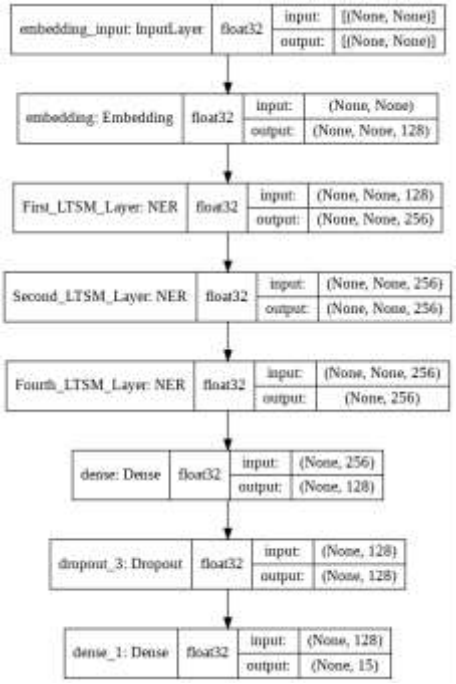


Fig. 22. LSTM model

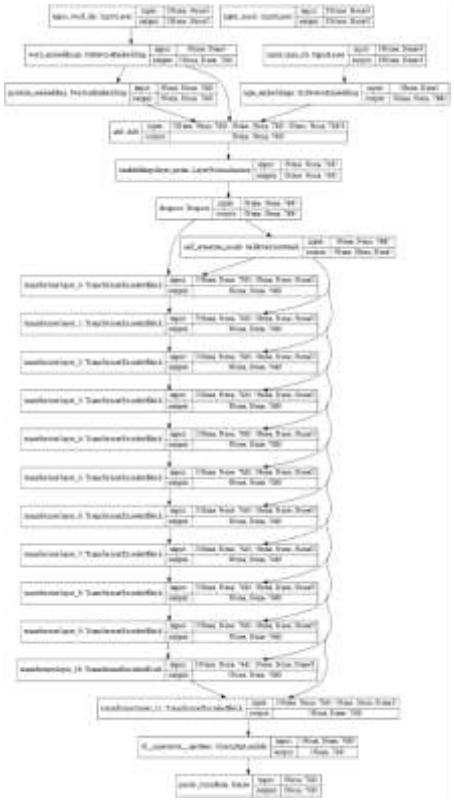
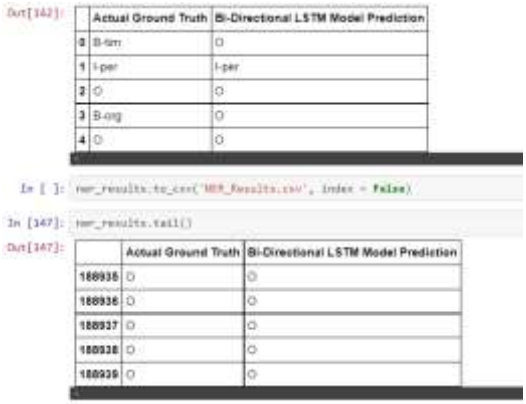


Fig. 23. BERT model



	Actual Ground Truth	Bi-Directional LSTM Model Prediction
0	B-tim	O
1	I-per	I-per
2	O	O
3	B-org	O
4	O	O

```
In [ ]: ner_results.to_csv('BDR_Results.csv', index = False)
In [147]: ner_results.tail()
```

Out[147]:

	Actual Ground Truth	Bi-Directional LSTM Model Prediction
188935	O	O
188936	O	O
188937	O	O
188938	O	O
188939	O	O

Fig. 23. Testing results

The main part of the whole process was creating the dataset, the model gave good results when all the tags of the dataset were evenly distributed else model gave bias prediction towards the tag which was more in number irrespective of accuracy. The models gave bias prediction when model achieved good prediction and also when model achieved average prediction, so it was found that model gave the wrong prediction that was due to dataset. It was found that creating the dataset was a more causal part followed by building the model

5. CONCLUSION

In this paper, we showed the masking of private data using Named Entity Recognition using Deep Learning concepts and how it can be used for achieving data anonymity. We talked about using various algorithms to achieve this and also portrayed the results achieved. Data is something we should always be very careful with. It describes us, helps us in finding things of similar interest like us but we must remember that it can also be used for manipulating how we think and function. This model recognizes the tags by their value and that value will be masked to generalized detail. In the dataset, we have many tags but we have limited the tags and taken only essential tags according to our application requirements. If there is any person, organization name or so on then the model will recognize it by its tags which will be masked as an output. We hope that our work will enhance the development of such tools soon.

6. REFERENCES

[1] Di Cerbo, Francesco & Trabelsi, Slim. (2018). Towards Personal Data Identification and Anonymization Using Machine Learning Techniques: ADBIS 2018 Short Papers and Workshops, AI*QA, BIGPMED, CSACDB, M2U, BigDataMAPS, ISTREND, DC, Budapest, Hungary, September, 2-5, 2018, Proceedings. 10.1007/978-3-030-00063-9_13.

[2] Vincze, Veronika, and Richárd Farkas. "De-identification in natural language processing." 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). IEEE, 2014.

[3] Medlock, Ben. "An Introduction to NLP-based Textual Anonymisation." LREC. 2006.

[4] Webster, Jonathan J., and Chunyu Kit. "Tokenization as the initial phase in NLP." COLING 1992 Volume 4: The 15th International Conference on Computational Linguistics. 1992.

[5] Manning, Christopher D., et al. "The Stanford CoreNLP natural language processing toolkit." Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations. 2014.

[6] Sherstinsky, Alex. "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network." Physica D: Nonlinear Phenomena 404 (2020): 132306.

[7] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[8] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," in IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, Nov. 1997, doi: 10.1109/78.650093.

[9] Vaswani, Ashish & Shazeer, Noam & Parmar, Niki & Uszkoreit, Jakob & Jones, Llion & Gomez, Aidan & Kaiser, Lukasz & Polosukhin, Illia. (2017). Attention Is All You Need.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." 2018 [Online]. Available: http://arxiv.org/abs/1810.04805

[11] Polikar R. (2012) Ensemble Learning. In: Zhang C., Ma Y. (eds) Ensemble Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-1-4419-9326-7_1

[12] L. K. Hansen and P. Salamon, "Neural network ensembles," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993-1001, Oct. 1990, doi: 10.1109/34.58871.

[13] Y. Liu, X. Yao, Ensemble learning via negative correlation, Neural Networks, Volume 12, Issue 10,1999, Pages 1399-1404, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(99)00073-8.

[14] MacKay D.J.C. (1995) Developments in Probabilistic Modelling with Neural Networks — Ensemble Learning. In: Kappen B., Gielen S. (eds) Neural Networks: Artificial Intelligence and Industrial Applications. Springer, London. https://doi.org/10.1007/978-1-4471-3087-1_37

[15] David H. Wolpert, Stacked generalization, Neural Networks, Volume 5, Issue 2, 1992, Pages 241-259, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(05)80023-1.

International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 4, July-Aug-2021, ISSN (Online): 2395-566X

Smart Automation Attendance System using Neural Networks

Satwik Ram Kodandaram
AI/NLP Scientist
Information Services Group,
Karnataka, India.
satwikram29@gmail.com

Kushal Honnappa
AI/NLP Scientist
Avkara/ Vmitis
Rajasthan, India..
kushal.h1999@gmail.com

Rishab Darshan S
Department of Computer Science,
Visvesvaraya Technological University,
Karnataka, India.
rishabds99@gmail.com

***Abstract*- A Traditional Attendance system will be a huge burden on teachers. After marking down, it manually, they have to upload it to some database to maintain the student's records. When there is a batch of students more than 500, doing it manually is imaginary. To resolve this problem, a smart and auto attendance management system is utilized. But authentication is a very important issue in this system. The smart attendance system is generally done with the help of biometrics. Face recognition is one of the best biometric methods to improve this attendance system. Being a prime feature of biometric verification, facial recognition is used in several applications like video monitoring. By utilizing this system, the problem of proxies and students being marked present even though they are not physically present in the class can be easily solved. The main implementation step used in this system is face detection and recognizing the detected face. This paper proposes a model for implementing an automated attendance management system using ANN and CNN. After this, the student's faces are mapped to their USN (University serial number) or ID. When the student's face is recognized and if they are present in the current location of the teacher, automatically the USN mapped to that student is marked as present in the database. In this model, the teacher needs to hosts the system through a web application or android application. While hosting the teacher's current location is taken. This model will be a successful technique to manage the attendance and records of students.**

Keywords- Students, teachers, attendance, face detection, neural networks.

I. INTRODUCTION

All colleges or schools have to track attendance in some way, so they can recognize which students are arriving late and which are always sharp on time and to maintain the record. The traditional method involved old-fashioned punch clocks, signatures on paper sheets, or some other kind of manual system that requires human oversight. If the attendance is taken manually, few problems may occur some of them are

- It is very difficult to handle huge students. For example a student's batch of more than 500 students.
- It is very time-consuming to call all the student's names and marktheir attendance.
- There will be chances of giving proxy (Fake) attendance, where it is difficult to handle the proxy.
- Now, what if we want to submit or send the attendance record to the university where exam ticket is issued based on a certain percentage of attendance, again from manual to data entry job need to be done which is again time-consuming.
- Inaccurate and subject to manipulation ('time theft')
- High possibility of human error.

A solution to the above system is the smart automation attendance system via face detection and using the current location. If the student location matches the current location of the teacher then the only student is marked as present in the database. Face detection is one of the important biometrics used in a variety of applications that identifies human faces in digital images.

Fig 1. Face Recognition.

Face detection can be done using neural networks which is part of artificial intelligence, where a machine can learn by itself without being explicitly programming. The important thing in this system is to train the machine with the student's images so that machine will learn by itself to detect the faces of students.

II. LITERATURE REVIEW

There are many systems developed in colleges and industries to keep track of attendance. But the performance and stability problems.

1. An attendance system using face detection in a classroom environment:
This System works perfectly fine when there is a fewer number of students. If there is a student batch of 200-500, multiple cameras must be used. Students have to focus on cameras where there might be chances of mixing up of faces and some students might not be captured.

2. Biometric-based System:
The Biometric- based systems take a unique part of the human body and are used for an attendance management system. Example Iris, retina scan. The iris recognition system is a useful system but the main drawback is it can cause eye injuries.

3. Bluetooth Based System:
This system is highly efficient and 100% proxy can be avoided. However, the system requires 8 connections active at a time. Bluetooth doesn't allow more than 8 connections this is due to the master and slave concept.

III. NEURAL NETWORKS

1. The Neuron:
In biology neurons consists of branches called Dendrites and a long tail called axon as shown in Figure 2.

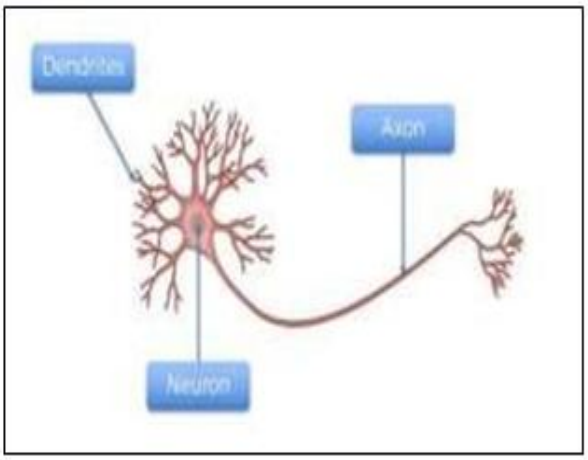


Fig 2. Neural Network system.

Dendrites are receivers and axons are transmitters. The key point to understand neurons is they are themselves useless. It is like a hand; five hands can do something. Dendrites are connected in neurons. Electrical signals are passed through neurons.

2. How to Represents neurons in Artificial Intelligence:
Technically neurons can be represented as below: Neuron is also called a node that receives an input signal and produces an output signal. These input signals come from other neurons called input layers. We also get input from hidden layer neurons. Here input signal is the independent variable in a model. Ex- age of a person, etc. we have to standardize and normalize the input variables. Output values can be continuous. Ex – price value.

Binary (Y or N Example whether a person will quite not.). Categorical example multiple values. Each signal consists of considerable weights. Weights are very important they are adjusted to learn by themselves. As shown in Figure 2, Neurons will decide what input signal is important based on these weights.

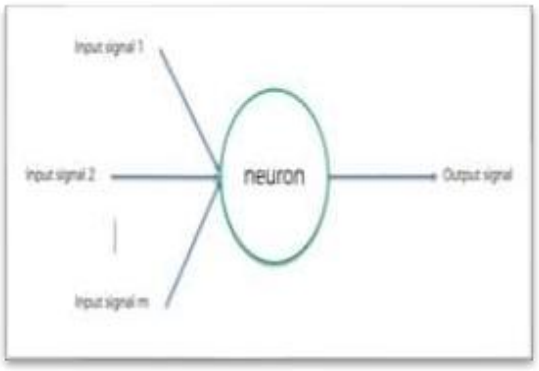


Fig 3. Neuron.

3. How does Neural Network works in Artificial Intelligence:
As shown in figure 3, in the first step all the weights are added and multiplied with independent variables. And a bias is also added in this step.

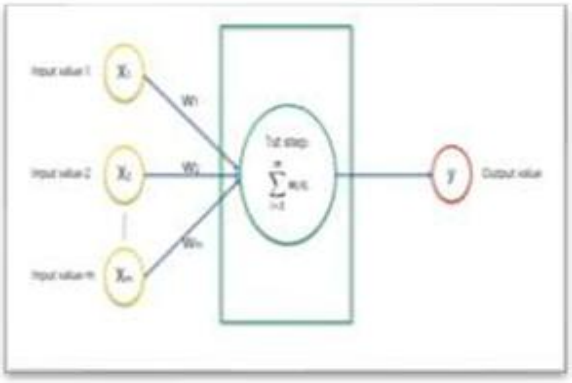


Fig 4. Neural networks in AI.

The basic formula for this is as follows

Y = w1*x1 + w2*x2 + w3*x3…+ wm*xm + Bias

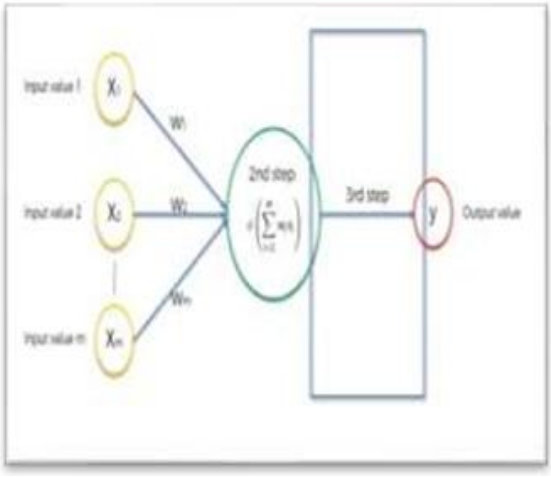


Fig 5. A neural network Activation function.

In the second step activation function is applied and in the third step, the signals are sent to the output layers. This complete step is called forward propagation.

The output of this is called as y predicted. This y predicted is compared with the actual y value and the weights are updated using the chain rule. This step is called Back Propagation

IV. FACE DETECTION USING CNN

1. Cerebral Cortex:
Human brain is divided into 4 parts. One of the parts is the cerebral Cortex. The Cerebral Cortex is the thin layer of the brain that covers the outer portion of the brain. Inside the cerebral cortex, we have the visual cortex.

2. Visual Cortex:
Visual Cortex is responsible for seeing the images. Suppose we are seeing a cat, that information is passed into our sensory organ eyes. Once the information is passed it will be passed into several neurons and then reaches the cerebral cortex region.

In the cerebral cortex we have a visual cortex, In the visual cortex, we have multiple layers say v1, v2, v3, and so on. These layers play a very important role. Suppose the v1 layer is responsible for finding the edge of the cat-like the eye edge of that, body edge of the cat and it is calculated by the v1 layer.

Then the information goes to the next layer that is v2, some more information is gathered in v2 like whether the cat is moving or sleeping, or any other object is present besides say dog and it is responsible to distinguish between cat and dog. Likewise, the information is passed to different layers and different information is gathered in different layers.

In each layer different operations are performed this is called filters in CNN. Finally, the image will be created in the visual cortex.

3. Convolution Neural Networks:
In the previous section we saw how the human brain recognizes the image. The same steps are followed in CNN.

4. The CNN has two important parts:

- **Feature Extraction:** In this part, the neural network will perform series of convolution and pooling operations. If we had a picture of a cat, it will recognize that the cat has two ears, 4four legs and it is sleeping.
- **Classification:** Here fully connected layer will treat as a classifier on the images. Suppose if two images are given one with a cat and the other with a cat, the model will classify the two different images with cat and dog.

The important steps involved in Convolution Neural Networks are:

5. Convolution:

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

6. Original Image:

1	0	-1
2	0	-2
1	0	-1

7. Image Filter:
In computer, images are represented in 0's and 1's. Here image consists of a 6x6 matrix and a filter consists of a 3x3 matrix. As discussed earlier v1 was responsible for finding the edge of the image. Likewise here filter represents the vertical edge filter. That is if we apply this filter to any image we can determine the vertical edge of the image. Multiply the first 3x3 matrix with the filter and add everything. Now jump by 1 and follow the same procedure. Once we come to end we should step one down

and continue the same procedure. The result of the multiplication of the matrix is a 4x4 matrix. This is called the Convolution operation.

0	-4	-4	0
0	-4	-4	0
0	-4	-4	0
0	-4	-4	0

8. Result:

These filters play a very important role. Here we just discussed the vertical edge filter. Likewise, we can have so many filters. Filters like finding the edge of the face, horizontal edge filter, and so on.

By looking at the result we can conclude a formula that image size n = 6 and filter size f =6.

$$\textbf{Result} = n - f + 1$$

$$\textbf{Result} = 6 - 3 + 1 = 4$$

If we look at the result, we are getting a 4x4 matrix but an image size 6x6 matrix. Here we are losing some of the information of the image. To avoid that we will apply the concept called Padding.

9. Padding:

$$n - f + 1 = 6$$
$$n = 6 - 1 + f$$
$$n = 5 + 3 = 8$$

Here we are getting n = 8 that is if we apply a padding p =1, that is one more row and column on an image we will get an image size of 8x8 matrix. On multiplying the image with a filter. We get the result as follows.

$$\textbf{Result} = n + 2p - f + 1$$

$$\textbf{Result} = 6 + 2 - 3 + 1 = 6$$

Here we get the resultant matrix of 6x6 and we are not losing any data. If we apply the padding formula we can retain the original size of the image.

We can apply n number of convolution and padding to train the images.

10. Max Pooling:

We saw how convolution and padding operations work. In max-pooling, one of the filters called the max-pooling filter is placed on the filters. Here we take the maximum value of the pixels and place it over a separate matrix.

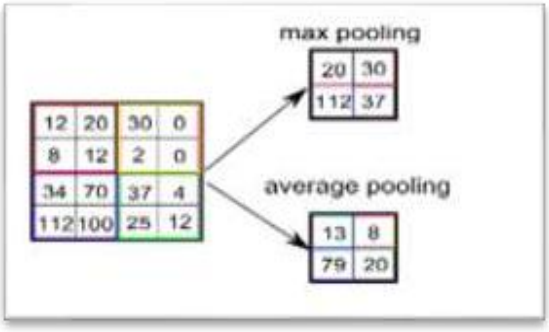


Fig 6. Max pooling.

Before we can barely recognize the face of the image. When max pooling is applied, we are taking only a high pixel value, so we can recognize the face of the image.

Flatten

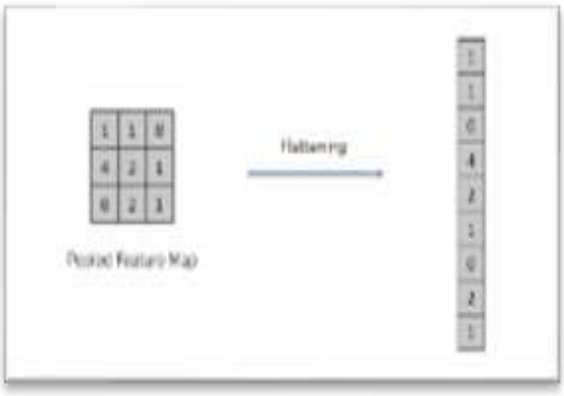


Fig 7. Pooled Feature Map.

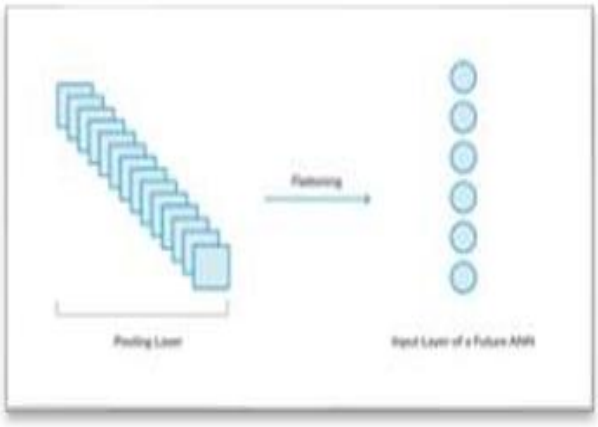


Fig 8. Pooling Layer.

11. Full Connection:

A full connection operation is applied at last. Where after flattening the outputs of flattened images are the input to the neural networks. As explained the working of neural network the same procedure is followed here

12. Back Propagation:

The output of the model is yhat, that is the predicted value. The actual output is y. Suppose yhat is 0 that is the

predicted value and the actual y value is 1. Here we try to see the difference between y and yhat, for that we use the loss function.

The loss function is calculated as:

$$\textbf{loss} = (y - yhat)^2$$

When we apply the loss function:

$$\textbf{Loss} = (1 - 0)^2 = 1$$

The predicted value is now matching with the actual value. Here the predicted value was the opposite. In the general case, we have to update the input weights so that the predicted value should match the output value. Here we define the loss function and our goal is to minimize the loss function so that yhat should match y. So in every epoch, we aim to update the value of the weight so that the loss value should decrease in every epoch. To achieve this we use optimizers, for Example, Gradient Descent.

13. MTCNN:

MTCNN- Multitasking Cascaded Convolutional Networks is a model for face detection which is available as a python library that is easy to use and install. It adopts a 3-stage neural network detector to detect the face of an object as well as bounding boxes with excellent precision. Not only it can capture the face but also it is capable of identifying facial landmarks like eyes, nose, and corners of a mouth.

Also, it can achieve a significantly faster rate of image processing of around 60-100 fps. Overall, this model is considered to be better suitable for real-time face detection of objects in images or videos [4].

Fig 9. Example MTCNN.

V. DATA AUGMENTATION

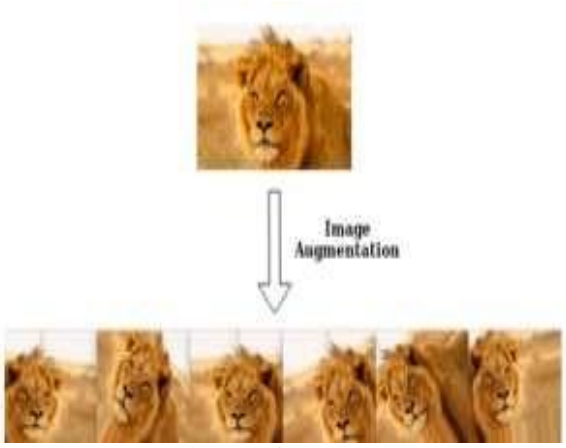


Fig 10. Data Augmentation.

Data Augmentation is one of the important steps in CNN. Data Augmentation helps us to increase the number of data available for training models, without actually collecting new data. Suppose if a have a cat image, is going input to the CNN, and the model will predict the output. With the help of data augmentation, we can transform the input cat image into different kinds of images.

Here still the output remains the same, we are just transforming the images to a different kind. Example flipping of image, Horizontal shifting, vertical shifting, and inverting of image. We can also zoom the image and send it to CNN. Whenever we have less data we can apply this technique to increase the images for training the model and to achieve the best accuracy.

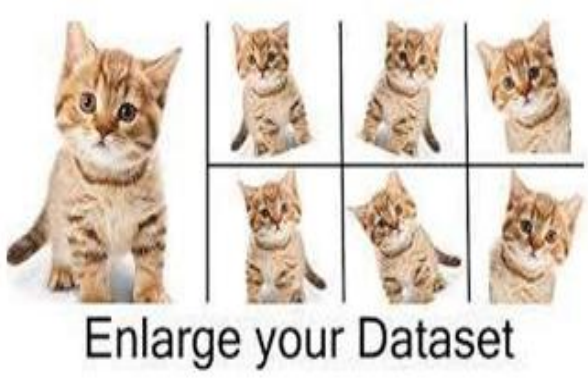


Fig 11. Data Augmentation.

If a student gives his face with a side angle our model should be able to predict the student. These transform images added are called invariance. Data Augmentation helps us to increase the images for training the model by almost 10x the original image. Suppose if we have 10

images, by applying this technique we can get 100 images thus improves the accuracy.

Fig 12. Data Augmentation generated.

The figure fig 12 shows the augmented images using our algorithm.

VI. PROPOSED MODEL

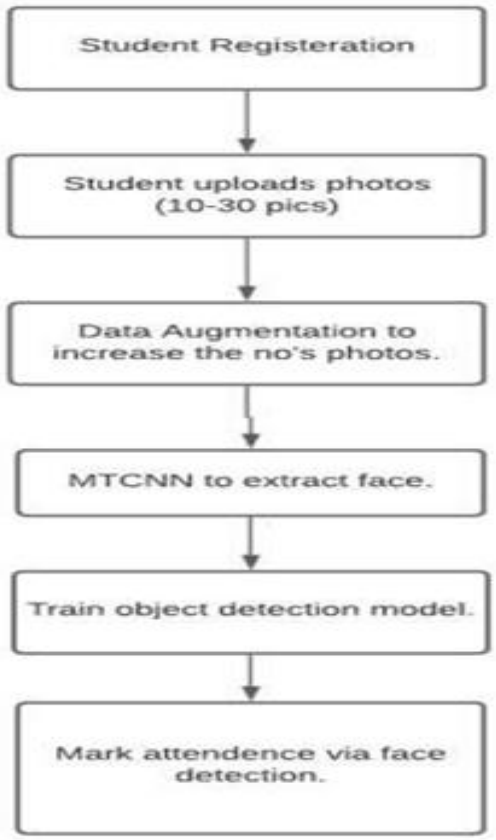


Fig 13. Proposed model flowchart.

As we all know how the technology is improved, everyone will have at least a basic smartphone, or students pursuing a degree will have a basic laptop. We will provide a website where students need to sign up to the website by providing all their information like USN, ID, and NAME, etc.

Here student image is compulsory where it is mapped into their ID or USN. Once the student is done with the sign-up process, they can log in to give their attendance by just giving their face identity (just like face unlock in the smartphone)

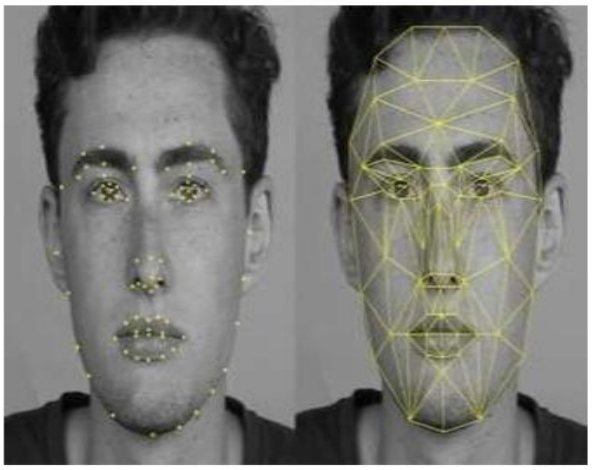


Fig 14. Face recognition.

1. How does it work?
On the website, there will be two login options

- Student login
- professor login

The subject professor needs to log in and hosts the attendance system where students can log in and give their attendance. Once the subject professor hosts the system, the professor's current location will be taken.

When students give their attendance by giving their face identity, if he/she is present in the current location then automatically image mapped to USN or ID will be marked as present in the database. If the student is not present in the current location of the professor then the student will be marked as absent in the database. Image recognition is done using Neural networks.

2. Methodology:
Image recognition is done using Tensor flow (Keras)
3. Techniques used:

- ANN(Artificial Neural Network)
- CNN(Convolution Neural Network)

3. The steps involved are:

- Initializing the model using Sequential class.

- Adding the Input layer that is images and applying the Convolution 2D method.
- Adding the Hidden layer and applying the activation function.
- Max pooling method is applied then.
- Flattening the image using the Flatten method.
- Adding the output layer and applying a suitable activation function. This is called the full connection method.
- Compiling the model.
- Predicting the model.
- Evaluating accuracy.
- Deploying the model using frameworks like Django web Frameworks or Flask

4. Web Developing steps:

- The frontend is done using Html, CSS, and JavaScript.
- Backend is done using Django or Flask.
- The database used Postgresql, SQL, etc.

VII. OBSERVATION

Fig 15. Home page.

Every time student data is trained again and again. If the student's image is recognized and if his/ her location is the same as the professor's current location then that student is marked as present in the database.

Fig 16. Face detection using MTCNN.

```
def draw_faces(filename, result_list):
        # load the image
        data = plt.imread(filename)
        # plot each face as a subplot
        for i in range(len(result_list)):
                # get coordinates
                x1, y1, width, height = result_list[i]['box']
                x2, y2 = x1 + width, y1 + height
                # define subplot
                plt.subplot(1, len(result_list), i+1)
                plt.axis('off')
                # plot face
                plt.imshow(data[y1:y2, x1:x2])
                filename = 'face'+str(i)+'.jpg'
                cv2.imwrite(filename, data[y1:y2, x1:x2])
        # show the plot
        plt.show()
```

```
draw_faces(filename, faces)
```

Fig 17. Face extraction using MTCNN.

Fig 18. Attendance system via face detection.

VIII. CONCLUSION

By using this system, we can overcome the current manual attendance system. 100% of proxy attendance can be reduced. Easy to maintain and evaluate student attendance for every subject.

REFERENCES

[1] M. Sajid, R. Hussain, and M. Usman, "A conceptual model for automated attendance marking system using facial recognition," Ninth International Conference on Digital Information Management (ICDIM 2014), Phitsanulok, 2014, pp. 7 -10.

[2] K. Puthea, R. Hartanto and R. Hidayat, "A review paper on attendance marking system based on face

recognition," 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, 2017, pp. 304-309.

[3] S. Bhattacharya, G. S. Nainala, P. Das and A. Routray, "Smart Attendance Monitoring System (SAMS): A Face Recognition Based Attendance System for Classroom Environment," 2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT), Mumbai, 2018, pp. 358-360.

[4] Xiang, J., & Zhu, G. (2017, July). Joint face detection and facial expression recognition with MTCNN. In 2017 4th international conference on information science and control engineering (ICISCE) (pp. 424-427). IEEE.

MACHINE LEARNING ALGORITHMS

Satwik Ram K(1va17cs047) and Pavan Kumar(1va17cs028)
Department: Computer Science
Sem: V
Gmail: satwikram29@gmail.com pavankumarn.17cs@saividya.ac.in

Abstract

We have seen how Artificial Intelligence has ruled the market. When we look at the deep coding of Artificial Intelligence, we will come across some of the mathematical analyses like principal component analysis (PCA), linear dimensional analysis (LDA), and so on. These concepts were introduced in the early 1900s and there was no much practical implementation because we didn't have enough data to implement. Now, these concepts are so trending in this modern era. This paper includes Why AI? What is the surge for AI? And some of the AI Algorithms and working of neural networks.

1.Introduction

Artificial Intelligence is an approach to make a computer, a robot, or a product think of how smart humans think. AI is a study of how the human brain thinks, learn, decide, and work when it tries to solve problems. And finally, this study outputs intelligent software systems. AI aims to improve computer functions that are related to human knowledge, for example, reasoning, learning, and problem-solving.

2.What is the surge for AI

Before discussing further AI, first, let's understand why there is a huge demand for AI. All the concepts of Data Science or AI were introduced in the early 1900′s, but there was no much practical implementation of these concepts. Post 1990′s the internet came into the picture and people started using the internet such that without the internet a human activity is incomplete. When we look at the data usage on an average:

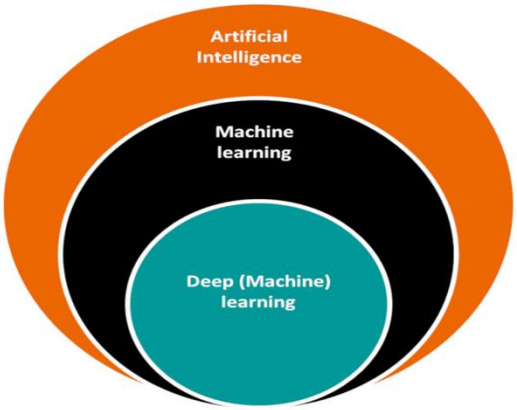
- A Human produces 25GB of IP traffic per month.
- A smart car will generate 2X that amount of data (50 GB)
- A smart hospital will generate 120X (3TB or 3,000 GB)
- A plane will generate 1,600X (40TB or 40,000 GB)
- A smart factory will generate 40,000X (1PB or 1,000,000 GB)
- A city safety system will generate 800,000X (50PB or 50,000,000 GB).

"And we're only talking about 2019!! "

We have produced tons of billions of data in past years now that we have the data and this data can be used to implement Artificial intelligence.

3.What is AI

AI can be understood as an ability to think and perform like humans without explicitly programming/ inducing knowledge into a machine artificially.



By looking at the image one can easily figure out that AI consists of in generally:

1. Machine learning
2. Deep learning (Neural Networks)

4.What is Machine Learning

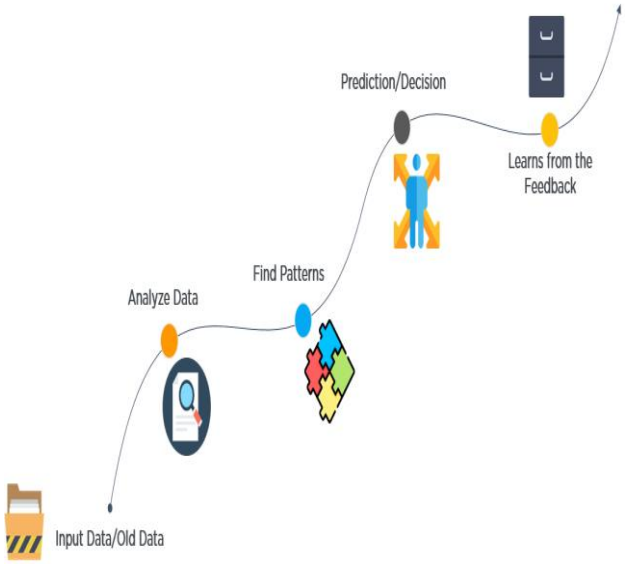
Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly.

So In Machine learning

1. We are trying to teach the Machine
2. Learn from the data
3. Follow the Instructions/Patters.

5.What does Machine Learning does



1. Input data/ old data
2. Analyze Data
3. Find patterns
4. Predictions/Decisions
5. Learn from the Feedback.

6.Why Python

Python is a general-purpose programming language that is often applied in scripting roles.

So, Python is a programming language as well as a scripting language. Python is also called an interpreted language.

Although we have so many other languages like c,c++, java, etc

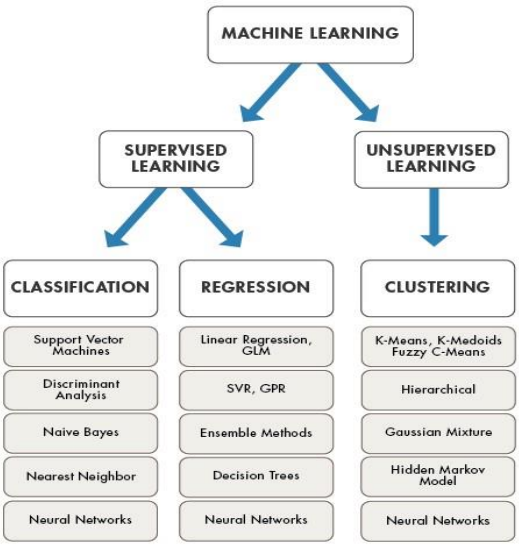
Why python is used in AI?

Because python has the following Advantages

1. It’s simple to learn. As compared to, C, C++, and Java the syntax is simpler.
2. Data handling capacity is great.
3. Open Source!

4. The capability of interacting with almost all third-party languages and platforms.

7.Machine Learning Algorithms



Supervised Learning:

In supervised learning, there will be labels that are the name given for Independent Variables

Unsupervised Learning:

In Unsupervised learning, there will be no labels and we don't have any idea about labels, we will start to identify and classify the labels and train the machine.

8.Regression

Regression analysis is a form of predictive modeling technique that investigates the relationship between a dependent and independent variable.

The above definition is bookish, in simple terms the regression can be defined as,

"Using the relationship between variables to find the best fit line or the regression equation that can be used to make predictions".

9.Simple Linear Regression

Linear regression models are used to show or predict the relationship between two variables of factors. The factor that is being predicted (the factor that the equation solves for) is called the dependent variable. The factors that are used to predict the value of the dependent variable are called the independent variables.

Good data does not always tell the complete story. Regression analysis is commonly used in research as it establishes that a correlation exists between variables. But correlation is not the same as causation.

Even a line in a simple linear regression that fits the data points well may not say something definitive about a cause-and-effect relationship.

The simple linear regression equation is represented like this:

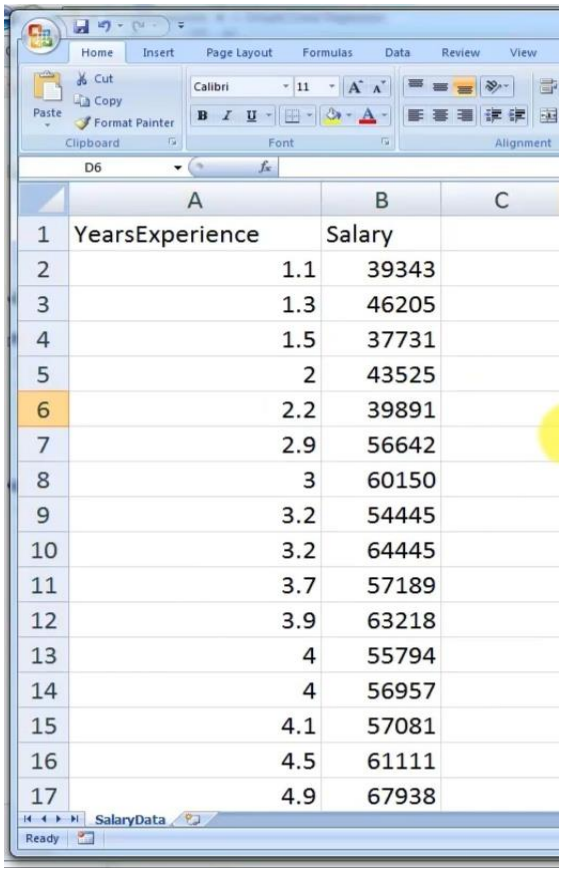
$$E(y) = (\beta 0 + \beta 1\ x).$$

The simple linear regression equation is graphed as a straight line.

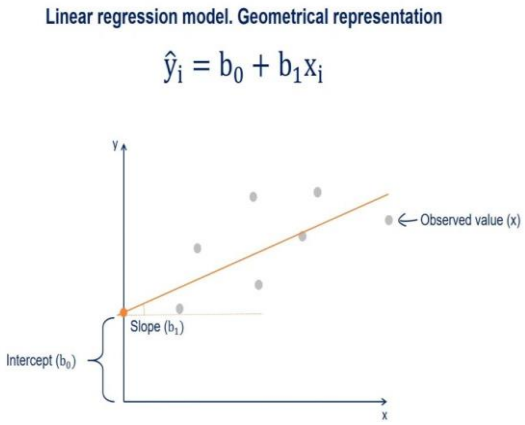
β0 is the y-intercept of the regression line.

β1 is the slope.

E(y) is the mean or expected value of y for a given value of x.



	A	B	C
1	YearsExperience	Salary	
2	1.1	39343	
3	1.3	46205	
4	1.5	37731	
5	2	43525	
6	2.2	39891	
7	2.9	56642	
8	3	60150	
9	3.2	54445	
10	3.2	64445	
11	3.7	57189	
12	3.9	63218	
13	4	55794	
14	4	56957	
15	4.1	57081	
16	4.5	61111	
17	4.9	67938	



One variable denoted x is regarded as the predictor, explanatory, or Independent variable.

Here Independent variable is Experience

The other variable denoted y is regarded as the response, outcome,
or dependent variable.

Here the dependent variable is Salary.

10.Multiple Linear Regression.

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable.

The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variable.

Explaining Multiple Linear Regression

Simple linear regression is a function that allows an analyst or statistician to make predictions about one variable based on the information that is known about another variable. Linear regression can only be used when one has two continuous variables—an independent variable and a dependent variable. The independent variable is the parameter that is used to calculate the dependent variable or outcome.

A multiple regression model extends to several explanatory variables.

Consider the Example

Profit	R&D Spend	Admin	Marketing	State
192,261.83	165,349.20	136,897.80	471,784.10	New York
191,792.06	162,597.70	151,377.59	443,898.53	California
191,050.39	153,441.51	101,145.55	407,934.54	California
182,901.99	144,372.41	118,671.85	383,199.62	New York
166,187.94	142,107.34	91,391.77	366,168.42	California

Y = b0 + b1*X1+ b2*X2+……bn*Xn

Here Profit is the Dependent variable

R&D Spend, Admin, etc are the independent variables on which profit is depending.

11.Polynomial Regression

Advantages of using Polynomial Regression:

Polynomial provides the best approximation of the relationship between the dependent and

independent variable.

A broad range of functions can be fit under it.

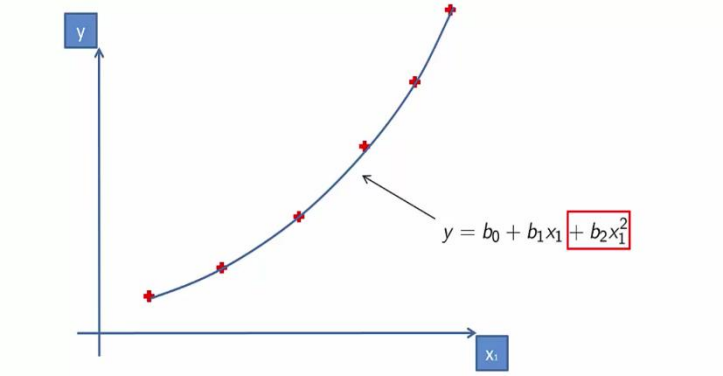
Polynomial fits a wide range of curvatures.

Disadvantages of using Polynomial Regression

The presence of one or two outliers in the data can seriously affect the results of the nonlinear analysis.

These are too sensitive to the outliers.

Also, there are unfortunately fewer model validation tools for the detection of outliers in nonlinear regression than there are for linear regression.



A curve in the polygraph

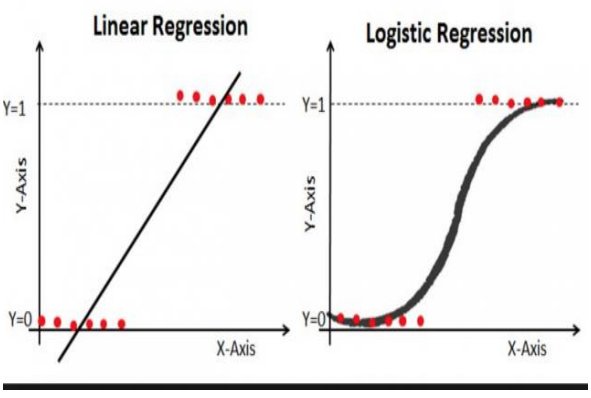
General formula is:

Y = b0 + b1*X1^2 + b2*X1^2 +…..bn*X1^n

12.Logistic Regression

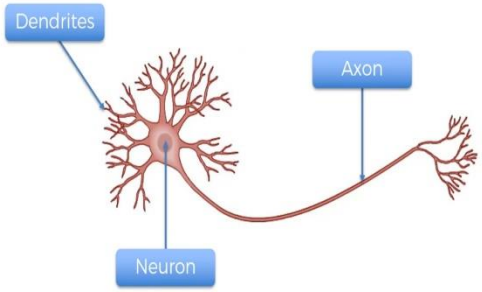
Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables.

Sometimes logistic regressions are difficult to interpret the Intellect us Statistics tool easily allows you to conduct the analysis, then in plain English interprets the output.



Neural Networks

13.The neuron



In biology, neurons consist of branches called Dendrites and a long tail called an axon. Dendrites are receivers and axons are transmitters. The key point to understand neurons is they are themselves useless. It is like a hand with five hands can do something. Dendrites are connected in neurons. Electrical signals are passed through neurons.

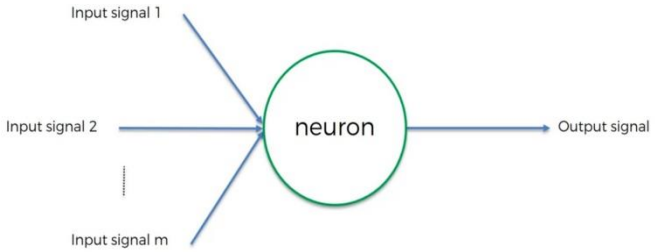
14.How to Represents neurons in AI

Technically neurons can be represented as below:

A neuron is also called a node which receives an input signal and produces an output signal. These input signals come from other neurons called input layers.

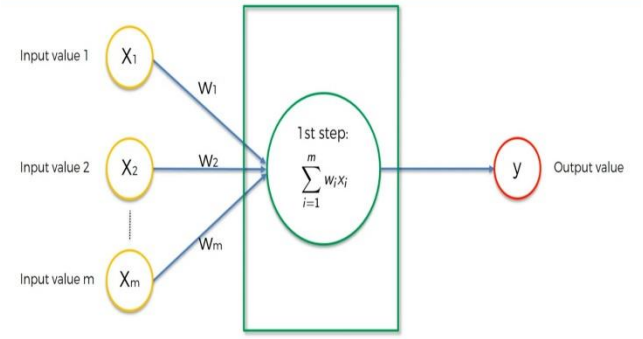
We also get input from hidden layer neurons. Here input signal is the independent variable in a model. Ex- age of a person, etc. we have to standardize and normalize the input variables.

Output values can be continues. Ex – price value. Binary(Y or N Example whether a person will quit or not .). Categorical ex-(multiple values). Each signal consists of considerable weights. Weights are very important they are adjusted to learn by themselves. Neurons will decide what input signal is important based on these weights.



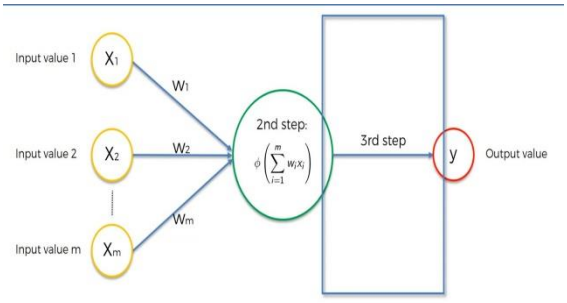
15.How do Neural Network works in AI

1.



In this step, all the weights are added and multiplied with independent variables.

2.



In the second step activation function is applied and in the third step, the signals are sent to the output layers.

16.Applications

1. In Automobiles **self-driving cars.**

2. In medicine, Machines can **recognize** the **disease** or **symptoms** or conditions related to the disease.

3. In Space, **Navigation Systems (Google maps)**

4. In Social media the **notification**, the **feeds** in the **timeline** all are of AI work.

5. In Real estate **AI camera** can be used to predict the damage in buildings etc.

6. In entertainment, A **robot dog.**

7. In Shopping, **personalized** shopping experience, **suggestions**, and **recommendations**.

8. And last but not the least ultimate technology which behaves like human **Sophia**

However, there is a **miss-use** of the **technology** also**,** Example of this **DeepNude**.