UNMASKING DECEPTIVE PROFILE: A NEURAL NETWORK APPROACH

Dr. D. Sunitha
Assistant professor
Department of Electrical and Electronics
Velammal Engineering College Chennai-600 066.

Dr. S. Premalatha
Associate Professor
Department of Electrical and Electronics
SRMIST
Chennai- 600 089

Kabilan D 113221051023
Student
Department of Electrical and Electronics
Velammal Engineering College Chennai – 600 066.

Arun K S 113221051008
Student
Department of Electrical and Electronics
Velammal Engineering College Chennai – 600 066.

Mahendhar V S 113221051030
Student
Department of Electrical and Electronics
Velammal Engineering College Chennai – 600066.

***Abstract* — Fake profile detection has become a crucial task in modern days because of the rise in fake accounts in social media which may be used to spread misinformation, conduct fraud activities, influence abnormal public opinion, sextortion, spam messages, cyber bullies and malicious activities and even more unethical activities using fake virtual identity. So, we have trained a Multi-Layer Perceptron (MLP) neural network which can outperform Machine Learning algorithms, Ensemble Learning algorithms, and even stacking of models. The MLP is trained using the dataset of features extracted from user profiles, including the number of times the username changed, the date on which the account was created, profile picture, length of username, number of words present in full name, full name length, name matching username, description length, external URL, private status, recent activity of them in platform by analyzing posts, followers, followings and likes for the posts. The dataset has features with binary values of 0s and 1s indicating whether a profile is fake or genuine. The MLP is trained using the Adam optimizer and the binary cross- entropy loss function, and its performance evaluation is done using accuracy and F1-score metrics. The results of the model show that the MLPs out-performs all the traditional Machine learning algorithms, with an accuracy of 95.32% and validation accuracy (Accuracy for an unknown dataset) of 91.43, the same way with a f1 score of 0.90625 These data results demonstrate the effectiveness of MLPs in identify the fake profiles in social media.**

Keywords— Fake profiles in social media, Machine learning techniques, Ensemble learning techniques, Neural networks, Perceptron, Multi-layered perceptron, social media analysis.

I. INTRODUCTION (*HEADING 1*)

In the modern era of social media, the conflicts for likes and shares are much higher which made influencers even public to follow unethical ways to get those. This leads to the creation of fake accounts on those platforms unidentified virtual accounts may influence the public in uncommon opinions, tempt fraudulent money from the public, and sextortion. Statistics show that Facebook has identified and removed 2.2 billion accounts from its platform which is very huge number. Even now Instagram and X (Twitter) haven't taken any action about these on their platforms that's where the neural network trained us and is very useful in classifying fake profiles and removing them from their platforms. Even a month ago Facebook CEO apologized in a US Senate hearing where the taking point was that every 2 minutes a Child is bought or sold on social media for these purposes mostly, they use fake profiles on platforms in which identity is hidden. No one could stop them from creating a fake profile but we identify them for our safety and our Children's.

II. LITERATURE SURVEY

A. *Deep learning By Ian Goodfellow, Yoshua Bengio and Aaron Courville*

This book covers fundamentals of deep learning, including NNs, CNNs, RNNs and more.

B. *Neural Networks and Deep Learning: A Textbook By Charu C. Aggarwal*

This book provides a thorough introduction to deep learning and neural networks, covering both theory and practical applications. It includes detailed explanation of various neural network architectures, including multilayer perceptrons (MLPs).

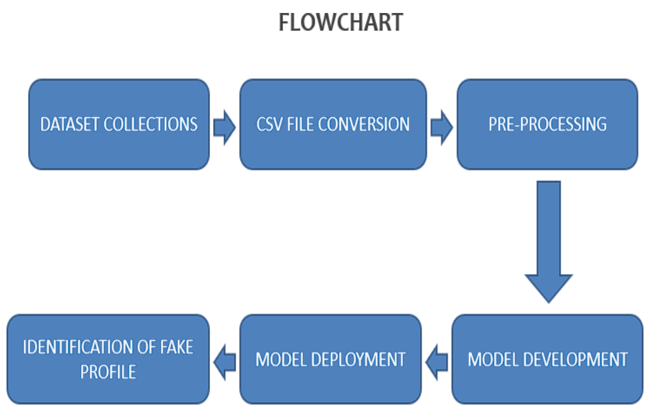
C. *Neural Networks: A Comprehensive Foundation By Simon Haykin*

This book covers fundamentals of neural networks, including multilayered perceptron(MLPs), radial basis function networks and more.

D. *Neural networks and Learning machines By Simon O.Haykin*

This book covers neural networks and learning machine with its methods of learning such as guided learning, non-guided learning and hit or reward based learning.

III. Block Diagram



IV. Methodology

Neural networks have proven themselves a remarkably potent instrument for tackling multi-attribute categorization challenges, especially in undertakings where the dataset encompasses a wealth of characteristics. Here's a simplified explanation of how neural networks work for multi-feature classification:

1. Input layer: The input layer consists of nodes, Each corresponding to a future in your data set. Total nodes is equal to total features we have in the data set.
2. Hidden layer: These are layers of nodes between the input and the output layers. When Each node in the hidden layer is connected to every node in the previous layer then it is known as a fully connected network and has a weight associated with each connection. These wigs are adjusted during training to minimize the error that is present in the network's prediction. The complexity of network depends upon the number of hidden layers in the network.
3. Output layer: The output layer is also made up of nodes which mostly correspond to the classification problem hence the output layers will very less in number as compared to other layers. If we have three class output functions the network would have three nodes on the output.
4. Activation function: Every node of the network is connected to another node through an activation function. Activation function is applied as weighted sum to the input before it reaches the next node. These activation functions introduce the linearity and non-linearity relationship between input and output vectors in a network.
5. Training: Networks are trained using a labeled or un-labeled dataset in our case network is trained using a labeled dataset. Often optimization algorithms are used to adjust the network's weight. We went for the Adam optimizer (Adaptive Moment Estimation) which is an extension of Stochastic gradient descent (SGD) optimization.
6. Prediction: After the training and evaluation of the network, it can be deployed to make forecasting on new datasets that should be unseen for it, this helps in better performance of the network.

V. Proposed System

The proposed system works on the principle of binary classification of nodes of the network where each node in the network is known as a Perceptron or neuron.

1. **Perceptron:** Fundamentally, a perceptron is a basic unit that can be used to implement artificial neural networks using binary classification. It comprises four major parts namely: inputs, weights, a weighted sum, and an activation function.

   a. Inputs: A perceptron accepts an array of input values which are typically real numbers in vector form. Each input has a weight associated with it that gives its relative significance in the model.

   b. Weights: These are actual numbers that are multiplied by their corresponding input values. This learning process occurs during training and determines how much importance is attached to the link between inputs and output.

   c. Weighted Sum: The weighted sum is obtained by multiplying each input with its respective weight, and then adding up the outcomes. After this happens, the resulting value undergoes an activation function which determines the output from this unit.

   d. Activation Function: This function takes in the weighted sums and returns as output what will be given out by perceptron machines. It introduces non-linearity into the network hence changing its behavior. Sigmoid function, step function or rectified linear unit (ReLU) function are some examples of commonly used activation functions for this purpose.

   e. Training: During the process of training, the perceptron trains its weights to get their optimal values by updating them based on the difference between the forecasted and real output. Typically, this is done through an optimization algorithm such as stochastic gradient descent (SGD).

   f. Limitations: How ever perceptron are effective in simple binary classification tasks but they have limitations. For instance, they can't learn non-linear decision boundaries and are not appropriate for any task that requires complicated decision-making.

g. Extensions: More complex models like MLPs and CNNs have been developed to overcome these drawbacks. These models have many layers of perceptron thereby enabling them to detect more intricate patterns in data.

We have used many perceptrons in a single layer, 64 on a single layer, and even layers up to 6 which helped to produce the desired accuracy. This type of combining many layers in a network is known as a Multilayered perceptron network (MLP). Our model consists of 6 layers including the input and output layers where 5 layers are trained by Relu (Rectified linear Unit) activation function the last layer (Output layer) consists single perceptron with Sigmoid activation which is highly useful in binary classification applications.

2. **Multilayered perceptron (MLP):** MLP is a kind of ANN that has many layers that are fed forward manner one by one. It is flexible and robust for several applications such as classification, regression and pattern recognition.

   MLP Structure:

   a. Input Layer: Every feature in the input data corresponds to a node on the Input layer which could be anything. The number of nodes in the input layer depends on the number of features in the dataset.

   b. Hidden Layers: These are nodes located between input and output layers. Each node in any hidden layer links to every other node in the previous layer and each connection carries its weight. The number of hidden node layers and several nodes in each hidden layer become parameters whose values may be adjusted depending on how complex a problem is.

   c. Output Layer: In classification problems, every class is assigned with a single node while for those related to regression, there's only one value per example. For any given data, network output signifies the probability that each category can be used to identify it; or more precisely, prediction for a regression problem.

   Activation Functions: In each hidden and output layer, a node applies an activation function to the weighted sum of its inputs. Popular activation functions include the sigmoid, hyperbolic tangent, and rectified linear unit (ReLU) functions. These functions introduce non-linearities, enabling the network to capture intricate data relationships.

   Training an MLP: The network undergoes training using a labeled dataset. Throughout this process, an optimization algorithm, such as gradient descent, fine-tunes the network's weights to minimize the discrepancy between its predictions and the actual labels. This technique, known as backpropagation, entails determining the gradients of the loss function with respect to the network's weights and adjusting the weights in a manner that reduces the loss.

   Prediction: Once the network is trained, it can be used to make predictions on new, unseen data. The input data is passed through the network, and the output of the network is the probability that each class is the correct classification for the input data or the predicted value in a regression problem.

3. **Activation Functions:** The weight that connects the two perceptrons together is known as the activation function. This function introduces the linear and nonlinear relationship between the input and output vectors. In our model, we have used ReLU (Rectified Linear Unit) activation and Sigmoid activation.

   a. Rectified Linear Unit (ReLu): It is a piecewise linear function that returns 0 for negative input and the same input value when it is greater than 0. It avoids the vanishing gradient problem and allows the network to learn quickly.

   The activation function of ReLU is:

   $$f(x) = \max(0, x)$$

   The derivative of function is given by:

   $$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

   b. Sigmoid function: As the name suggests it is an S-shaped curve which maps the input value to any value between 0 and 1. It is mostly used in the output layer for the binary classification where probability of class considered.

   The activation function of sigmoid is:

   $$f(x) = \frac{1}{1 + e^{-x}}$$

   The derivative of the function is given by:

   $$f'(x) = f(x).(1 - f(x))$$

4. Optimizer: In our model we have used Adam optimization algorithm which update the weights neural network while training. It is the extended version of stochastic gradient descent (SGD) optimization algorithm.

Adam optimization, a blend of AdaGrad and RMSProp, capitalizes on the strengths of both. It borrows from AdaGrad by customizing the learning rate for each parameter according to its historical gradients. However, it also draws from RMSProp by incorporating exponential moving averages of gradients and squared gradients, enabling it to dynamically adapt the learning rate based on the optimization's current state.

The key equation for Adam optimizer as follows:

a. Exponential moving averages of gradient:

$$M_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$V_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

b. Bias correction:

$$\bullet\ \hat{m}_t = \frac{m_t}{1-\beta_1^t}$$
$$\bullet\ \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

c. Update rule:

$$\bullet\ \theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t$$

5. Epoch: Epoch represent how many times the model sees the dataset. When we set epoch to 10 it sees the datasets 10 times and when we set it to 20 model sees the datasets 20 times.

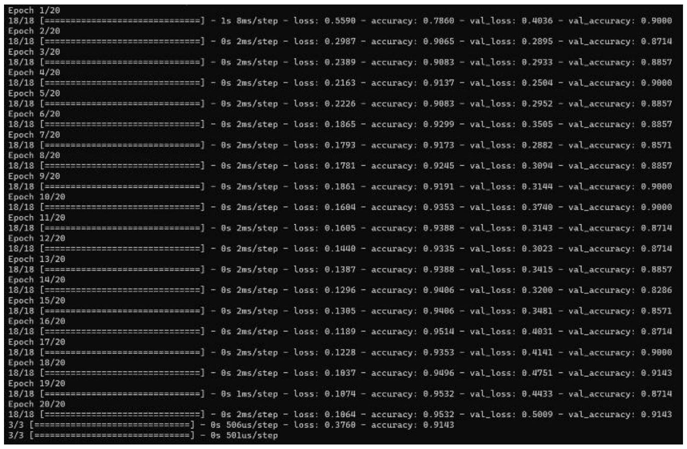
   a. Why 20 epochs: The choice of number epochs depends on various factors, including the complexity of the problem, the size of the dataset, the architecture of the model. For fake profile detection we initialize from 0 to 20 where on 20 epochs it reached its saturation.

   b. Monitoring Performance: During training, it's necessary to observe the model's performance on a validation dataset to avoid overfitting. You can do this by calculating metrics such as accuracy, precision, recall, and F1-score on the validation set after each epoch. When the performance on validation set is starts decreasing it is an alarming sign that model overfits on the datasets. Hence, further increase in the epochs can only cause decrease in the performance.

   c. Early stopping: To automatically stop training when the performance starts to degrade, you can use a technique called early stopping. It monitors the performance on validation set after each epoch when the performance decreases it automatically stops which is the early stopping method.

VI. Output of the Model

The model is trained to produce an output of 95.32% and validation accuracy of 91.43% and produced an F1 score 0.90625 which is much higher for a multilayered perceptron network. The below image clearly shows the output accuracy of the network which outperformed all the regular machine learning and ensemble learning techniques.



```
Epoch 1/20
18/18 [==============================] - 1s 8ms/step - loss: 0.5590 - accuracy: 0.7860 - val_loss: 0.4036 - val_accuracy: 0.9000
Epoch 2/20
18/18 [==============================] - 0s 2ms/step - loss: 0.2987 - accuracy: 0.9065 - val_loss: 0.2895 - val_accuracy: 0.8714
Epoch 3/20
18/18 [==============================] - 0s 2ms/step - loss: 0.2389 - accuracy: 0.9083 - val_loss: 0.2933 - val_accuracy: 0.8857
Epoch 4/20
18/18 [==============================] - 0s 2ms/step - loss: 0.2163 - accuracy: 0.9137 - val_loss: 0.2504 - val_accuracy: 0.9000
Epoch 5/20
18/18 [==============================] - 0s 2ms/step - loss: 0.2226 - accuracy: 0.9083 - val_loss: 0.2952 - val_accuracy: 0.8857
Epoch 6/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1865 - accuracy: 0.9299 - val_loss: 0.3505 - val_accuracy: 0.8857
Epoch 7/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1793 - accuracy: 0.9173 - val_loss: 0.2882 - val_accuracy: 0.8571
Epoch 8/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1781 - accuracy: 0.9245 - val_loss: 0.3094 - val_accuracy: 0.8857
Epoch 9/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1861 - accuracy: 0.9191 - val_loss: 0.3144 - val_accuracy: 0.9000
Epoch 10/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1604 - accuracy: 0.9353 - val_loss: 0.3740 - val_accuracy: 0.9000
Epoch 11/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1605 - accuracy: 0.9388 - val_loss: 0.3143 - val_accuracy: 0.8714
Epoch 12/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1440 - accuracy: 0.9335 - val_loss: 0.3023 - val_accuracy: 0.8714
Epoch 13/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1387 - accuracy: 0.9388 - val_loss: 0.3415 - val_accuracy: 0.8857
Epoch 14/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1296 - accuracy: 0.9406 - val_loss: 0.3200 - val_accuracy: 0.8286
Epoch 15/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1305 - accuracy: 0.9406 - val_loss: 0.3481 - val_accuracy: 0.8571
Epoch 16/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1189 - accuracy: 0.9514 - val_loss: 0.4031 - val_accuracy: 0.8714
Epoch 17/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1228 - accuracy: 0.9353 - val_loss: 0.4141 - val_accuracy: 0.9000
Epoch 18/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1037 - accuracy: 0.9496 - val_loss: 0.4751 - val_accuracy: 0.9143
Epoch 19/20
18/18 [==============================] - 0s 1ms/step - loss: 0.1074 - accuracy: 0.9532 - val_loss: 0.4433 - val_accuracy: 0.8714
Epoch 20/20
18/18 [==============================] - 0s 2ms/step - loss: 0.1064 - accuracy: 0.9532 - val_loss: 0.5009 - val_accuracy: 0.9143
3/3 [==============================] - 0s 506us/step - loss: 0.3760 - accuracy: 0.9143
3/3 [==============================] - 0s 501us/step
```

Accuracy: This is a measure of how many predictions the model got right out of the total number of predictions. The model perfectly predicted 95.32% of the samples in the training dataset.

Validation Accuracy: This is the accuracy of the model on a separate validation dataset that was not used during training. The validation accuracy means that the model perfectly predicted 91.43% of the samples in the validation dataset.

F1 Score: The F1 score is a measure of a model's delicacy on a double bracket problem. The harmonious mean of perfection and recall is reckoned, whereby perfection is calculated as the proportion of true positive prognostications out of all positive prognostications made, and recall reflects the bit of factual positive cases that were rightly prognosticated as similar. An F1 score of 0.90625 indicates that the model has a good balance between perfection and recall.

Interpretation:

The model's robust training accuracy (95.32%) reflects its adeptness at learning the nuances within the training data. The marginally lower validation accuracy (91.43%) signifies its capability to extrapolate well to novel, unobserved data. The impressive F1 score (0.90625) underscores its ability to achieve a harmonious balance between precision and recall, a crucial aspect for binary classification.

VII. Comparison Between Previous Developed Model

The previous developed models using machine learning and ensemble learning are outperformed by multilayered perceptron network. The main advantage of multilayered network is new network can be added whenever we found a new trend in fake profiles in social media. The datasets could be visited and revisited multiple times in using epochs of network.

Attributes	***Traditional machine learning methods***	***Multilayered perceptron network***
Adding new features	Adding new layer is impossible but some hyper-parameter tuning can be done by changing or adding new features in datasets.	Easy to add new feature as a new layer in multiple layered perceptron.
Accuracy	The produced accuracy is 86%	The produced accuracy is 90%. There is no much increase in accuracy but model often train on that dataset.
Epochs	The model visits the datasets only one time and recognize all the patterns in that datasets.	We can control how many times the model visits datasets using epochs.
Hyper-parameter tuning	Traditional machine learning methods have very few hyper-parameter tuning methods.	Hyper-parameter tuning methods wide and versatile in multilayered neural networks.
Non-linear relationsh ips	Non linear relationships in Random forest and LightGBM can be done but it requires multiple trees and trees with greater depth.	MLPs can easily capture all non-linear features between input vector and output vector.
Feature representa tion	These methods need more manual feature engineering for greater performance.	MLPs can automatically learn main features from data.

VIII. Conclusion

The Multilayer Perceptron (MLP) network has proven to be a highly effective tool for detecting fake profiles. With an accuracy of 95.32% and a validation accuracy of 91.43%, the model has proven its capability to grasp intricate data patterns and effectively adapt to unfamiliar data.

The model's high accuracy implies that it has effectively learned the core patterns within the data and can make precise forecasts for new, unobserved profiles. This is a significant achievement, as fake profile detection is a challenging task that requires the model to identify subtle differences between genuine and fake profiles. The validation accuracy of 91.43% suggests that the model is adept at generalizing to fresh, unfamiliar data, a critical aspect for real-world applications. This indicates that the model isn't excessively fitting to the training data and can accurately forecast profiles it hasn't encountered previously.

IX. Future Advancement

A. *Sentimental analysis of profile description:*

1) Sentiment analysis, a technique used to determine the emotional tone of text, can be incorporated into the fake profile detection model to analyze profile descriptions.

2) By identifying suspicious or unusual language patterns, such as overly positive or negative sentiments, the model can better discern fake profiles.

3) Sentiment analysis can be implemented as an additional feature, complementing the existing model's capabilities and potentially improving its accuracy.

B. *Fine tuning for emerging trends in fake profiles:*

1) Fake profiles are dynamic and evolve over time, often resulting in new trends. To ensure the model remains effective, it should be updated to detect these emerging trends.

2) Regularly updating the model with fresh data and refining its ability to identify emerging patterns in deceptive profiles is imperative.

3) Approaches such as transfer learning and active learning can be utilized. Transfer learning involves starting with a pre-trained model and adjusting it with new data, while active learning entails iteratively retraining the model with a small set of labeled data and then utilizing the model to label additional data.

4) These techniques help the model adapt to new trends without starting from scratch, ensuring its continued effectiveness in detecting fake profiles.

References

[1] Sarah Khaled, Neamat El-Tazi, Hoda M. O. Mokhtar "Detecting fake accounts on social media" 2018 IEEE International Conference on Big Data on 10-13 December 2018 Seattle, WA, USA.

[2] Samuel Delgado Munoz, Edward Paul Gullen Pinto, "A dataset for the detection of fake profiles on social networking services" 2020 International Conference on computational science and computational intelligence (CSCI) on 16-18 December 2020 Las Vegas, NV, USA.

[3] Pradeep Kumar Roy, Shivam Chahar, "Fake profile detection on social networking websites: A comprehensive Review" IEEE Transactions on Artificial Intelligence on 2020 December.