Text Data De-Anonymization

Sanjay Chari, PES1201700278
Department of CSE
PES University
Bangalore, India
sanjaychari2xy2@gmail.com

Nishanth Shastry, PES1201700115
Department of CSE
PES University
Bangalore, India
nishshastry@gmail.com

Ashwini Joshi
Faculty, Department of CSE
PES University
Bangalore, India
ashwinimjoshi@pes.edu

***Abstract*—Social media has taken over today's world. Social media sites contain millions of users. Along with their public profile, users are allowed to post content under the guise of anonymity. A tool capable of classifying a given post as originating from a given source would be beneficial in this aspect.**

In this paper, we explore the feasibility of such a tool using Natural Language Processing methods in conjunction with Machine Learning and Deep Learning models. Twitter data, primarily the tweet data of the top 50 most popular Twitter users (Donald Trump, Ariana Grande, etc.) is used as the data set. We attempt to analyze their textual pattern to classify the tweets as originating from a given profile. Various learning techniques have been applied in our model including Logistic Regression, Naive Bayes Classification and Stochastic Gradient Descent. Our model utilises an ensemble model of both Neural Networks (including a hybrid model) and Single Value Decomposition. Finally we report our performance metrics and results of the model.

***Index Terms*—Natural Language Processing, Anonymization, Text De-Anonymization, Machine Learning, Deep Learning, Classification, Twitter, Ensemble Learner, Neural Networks, Hybrid Model, Multiclass Classification, Social Media, Word2Vec**

I. Introduction

Social Media has become a ubiquitous resource, with everyone with a connection to the internet having the ability to access it. With this explosion, there are millions of users sending messages simultaneously. In several scenarios, users can be attacked while posting information under a publicly viewable identity. Predatory users can send messages consisting of foul language, inappropriate content, etc. This can even go as far as attacking using harmful viruses, performing illegal transactions like data mining, etc. Various websites can also be quite a nuisance by tracking the user for target advertisement purposes, content campaigns and general spam. Hence, it is crucial for users require a way to voice their opinions without giving away their online identity.

An indispensable tool that provides a solution to these issues is user anonymization. Now, sensitive users can post their thoughts/views without fear of identification for whatever personal reason. Other users can see the content posted but not the identity of the host. As with any tools, prevalent misuse can cause a myriad of problems both for other users and the social media network itself. Now users can post lewd/foul messages, hate speech, etc. and just do general inappropriate activity and malpractice that would not be tracked to their real persona. The same qualities that make anonymization helpful can be used equally against it.

It hence becomes necessary to develop a method/model to try and predict the identity of the anonymous poster. Future corrective action can then be taken if necessary. In some cases de-anonymization or identification can be a great help to the anonymous individual as well. For these reasons, we attempt to develop a software that would be capable of classifying a given instance of textual data as originating from a known host, using natural language processing, along with machine learning and deep learning models. Specifically, we utilize machine learning methods like Naive Bayes, Logistic Regression, and Stochastic Gradient Descent. We also utilize a neural network model and a hybrid model for our task.

II. Ethics of De-anonymization

As with any method of identification, the act of de-anonymization can be argued as a breach of privacy. Just the fact that some software is identifying an anonymous user, whether successfully or not, can lead to various security concerns. Before we move on to our approach, we would like to put this issue at ease.

As there is no conclusive right answer, the only way to proceed is to weigh the pros and cons and make a judgement call as to whether more good than bad will occur. The process of de-anonymization of anonymous users can bring a variety of benefits. On the smaller scale, it can help remove all sorts of hate speech, inappropriate content, etc. and generally make the social media platform a cleaner place. On the larger scale, it can even go as far as to prevent various illegal transactions, help identify threats, breaches, etc. that are posted online, etc. The major drawback is that the software can be used for unethical activities that can cause some discomfort for those that are identified.

We firmly believe that our approach of de-anonymization is beneficial and this justifies the ethics of the approach.

III. Problem Statement

The problem statement of our task mainly is to be able to identify or atleast predict with a good accuracy, the identity of the poster of an anonymous piece of text. It mainly involves multi-class classification as well as creating a suitable learner. The data-set that we utilize consists of tweets from the top 50 twitter profiles. Using machine learning and deep learning methods, our objective is to build classifiers that take a tweet as input and identify the twitter profile from which it originated.

IV. Literature Survey

A thorough research of the following research papers was done before constructing the proposed model. A brief summary of each research paper is mentioned below for further research.

1. De-anonymizing Web Browsing Data with Social Networks, 2017 ACM. ISBN TDB - Jessica Su, Stanford University; Ansh Shukla, Stanford University; Sharad Goel, Stanford University; Arvind Narayanan, Princeton University

Summary: Each person has a distinctive social network, and thus the set of links appearing in one's feed is unique. Assuming users visit links in their feed with higher probability than a random user, browsing histories contain tell-tale marks of identity. They formalize this intuition by specifying a model of web browsing behavior and then deriving the maximum likelihood estimate of a user's social profile.

2. Fast De-anonymization of Social Networks with Structural Information. Data Sci. Eng. - Shao, Y., Liu, J., Shi, S. et al.

Summary: They propose a fast and effective seedless network de-anonymization approach simply relying on structural information, named RoleMatch. RoleMatch equips with a new pairwise node similarity measure and an efficient node matching algorithm.

3. Optimal Active Social Network De-anonymization Using Information Thresholds - F. Shirani, S. Garg, E. Erkip

Summary: In this problem, an anonymous victim visits the attacker's website, and the attacker uses the victim's browser history to query her social media activity for the purpose of de-anonymization using the minimum number of queries. A stochastic model of the problem is considered where the attacker has partial prior knowledge of the group membership graph and receives noisy responses to its real-time queries.

4. De-identification in Natural Language Processing - Veronika Vincze, Richard Farkas

Summary: They discuss the questions of de-identification related to three NLP areas, namely, clinical NLP, NLP for social media and information extraction from resumes. They also illustrate how de-identification is related to named entity recognition and we argue that de-identification tools can be successfully built on named entity recognizers.

5. An introduction to NLP-based textual anonymization - Ben Madlock

Summary: They introduce the problem of automatic textual anonymisation and present a new publicly-available, pseudonymised benchmark corpus of personal email text for the task, dubbed ITAC (Informal Text Anonymisation Corpus). They discuss the method by which the corpus was constructed, and consider some important issues related to the evaluation of textual anonymisation systems. They also present some initial baseline results on the new corpus using a state of the art HMM-based tagger.

V. Dataset

The dataset we chose for our task can be found at:

https://drive.google.com/drive/folders/11w4geFB6p17hFlWseBpHJQbhARINvTOc.

This dataset consists of 50 folders, with each folder containing 4 files : tweets.csv, images.csv, videos.csv, and analysis.pdf. Each folder represents a unique twitter profile. These are the top 50 most popular twitter profiles. The reason we chose this dataset for our task is, the user profiles had specific unique tweeting patterns, that could be used to build an accurate classifier model using machine learning and deep learning techniques.

Since we are interested in the NLP aspect of the dataset, we combined all the data from tweets.csv into one single file dataset.csv, with the label for a given tweet as the twitter handle that posted the tweet.

We ran classifiers on the data from dataset.csv.

VI. Pre Processing

To correctly analyze the textual patterns in the tweets, we performed pre processing on our dataset.

Firstly, our dataset was present in the form of 50 folders, with each folder containing 4 files : tweets.csv, images.csv, videos.csv, and analysis.pdf. Each folder represents a unique twitter profile. As we were only interested in the Natural Language Processing aspect, we created a csv file, with each row in the csv file containing the the tweet, twitter user id, twitter username, and a numeric label that we assigned programmatically to the user.

After this, we performed the traditional pre processing steps on each tweet to extract meaning information. The steps performed are listed below.

1. Lowercasing : Text often has a variety of capitalization reflecting the beginning of sentences, proper nouns emphasis. The most common approach is to reduce everything to lower case for simplicity. For this reason, we have utilized this approach in our methodology.

Eg: Canada, CANADA, and CaNAda will all become canada after lowercasing

2. Tokenization : Tokenization describes splitting paragraphs into sentences, or sentences into individual words. We tokenized each tweet into words, and then removed words which would hinder our classifcation task.

Eg: ”This is our NLP project.” will be tokenized into ’This’, ’is’, ’our’, ’NLP’, ’project’, ’.’ respectively.

3. Stop Word Removal : A majority of the words in a given text are connecting parts of a sentence rather than showing subjects, objects or intent. Word like “the” or “and” can be removed by comparing text to a list of stop word. We also removed symbols like ”@”, which would hinder the classification process.

Eg: ’This’, ’is’, ’a’, ’sample’, ’sentence’ will become ’This’, ’sample’, ’sentence’ after stop word removal

4. Lemmatization : Lemmatization refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma. We utilized lemmatization to extract the root word from the tokenized tweets, to make our classifier training process more efficient. We compared the results generated by lemmatization and stemming, and observed that stemming gave poor results when compared to lemmatization. For this reason, we utilize lemmatization for our process.

Eg: ’Feet’ becomes lemmatized to ’foot’ as its the plural form of the word foot.

VII. Methodology

1. N - Grams

Simply speaking, an N-gram is a sequence of N words. N-grams alone aren’t very useful. When we assign a probability to the occurrence of an N-gram or the probability of a word occurring next in a sequence of words, there are several applications.

Our preliminary analysis began with the generation of the most frequent unigrams and bigrams for each of the 50 twitter profiles. These can tell us about the set of words that are most frequently used by that individual.

Eg: ’arianagrande’:
Most correlated unigrams - thankunext, love
Most correlated bigrams - thank thankunext, love sm

2. Word Cloud

A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it’s mentioned within a given text and the more important it is. Word clouds are ideal ways to pull out the most pertinent parts of textual data, from blog posts to databases. They can also help business users compare and contrast two different pieces of text to find the wording similarities between the two.

We also constructed a Word Cloud as part of our initial analysis. This helped us to identify some key words for individuals which can help later in their identification.

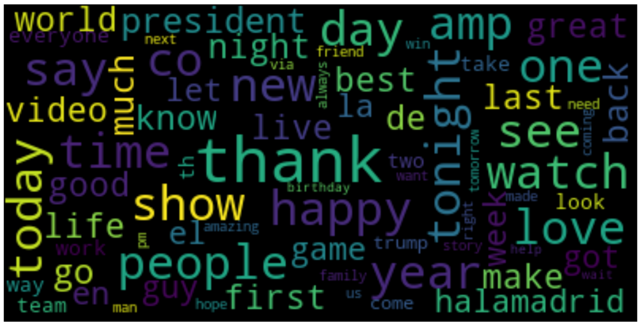


Fig. 1. Sample Word Cloud

3. Logistic Regression

A logistic regression model is a kind of parametric classification model that has a certain fixed number of parameters that depend on the number of input features. These models output a categorical prediction

In Logistic Regression, we don’t directly fit a straight line to our data like in linear regression. Instead, we fit a S shaped curve, called Sigmoid, to our observations.

We utilised Logistic Regression as one of multiple classification models to try to find the best one to add to our hybrid classifier. We chose this classifier as it requires low training time but still reports comparable accuracy with respect to other models.

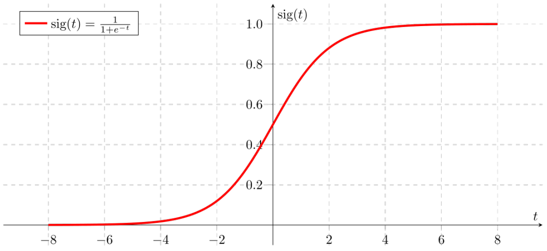


Fig. 2. Logistic Regression

4. Naive Bayes Classification

Conditional probability is the likelihood of an event A to occur given that another event that has a relation with event A has already occurred. A joint probability is the probability of two events occurring together

Naive Bayes is a supervised learning algorithm for classification given the values of features. Naive Bayes classifier calculates the probability of a class a set of feature values using Bayes' theorem.

We utilised Naive Bayes Classification as one of multiple classification models to try to find the best one to add to our hybrid classifier. We chose this classifier as it requires low training, and is a baseline classifier for Natural Language Processing classification tasks.

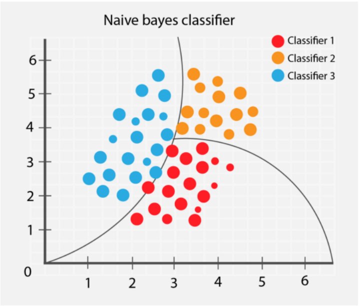


Fig. 3. Naive Bayes Classifier

5. Stochastic Gradient Descent

Gradient Descent is a very popular optimization technique in Machine Learning and Deep Learning. A gradient is basically the slope of a function. The more the gradient, the steeper the slope. Gradient Descent can be described as an iterative method which is used to find the values of the parameters of a function that minimizes the cost function as much as possible.

The word 'stochastic' means a system or a process that is linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration.

Suppose, you have a million samples in your dataset, so if you use a typical Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima is reached. Hence, it becomes computationally very expensive to perform. This problem is solved by Stochastic Gradient Descent. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration.

We utilised Stochastic Gradient Descent as one of multiple classification models to try to find the best one to add to our hybrid classifier. This classifier requires a longer training time, but promises good performance metrics. This is why we decided to choose this

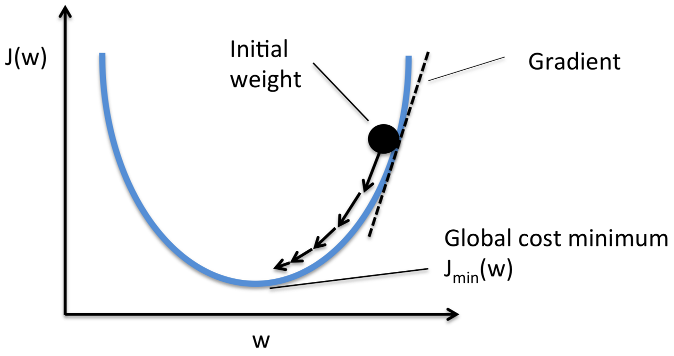


Fig. 4. Stochastic Gradient DescenGradientt

6. Neural Network

A Neural Network is a computer program that operates similarly to the human brain. The neural network itself has many small units called neurons, these neurons are grouped into several layers. Layers are columns of neurons that are connected to each other through their neurons. Each neuron is connected to another layers' neuron through connectors called weighted connections. A neuron takes the value of a connected neuron (in their layer) and multiplies it with their connections' weight. The sum of all the connected neurons is the neurons' bias value. Essentially, the network is like a filter through all the possibilities so, the computer can come up with the correct answer.

When we modelled our approach with just a regular neural network, we found that the model was pretty good and gave a good accuracy.

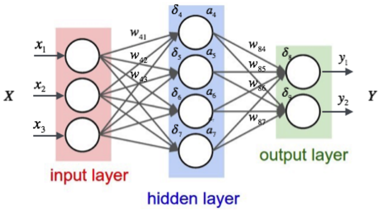


Fig. 5. Neural Network

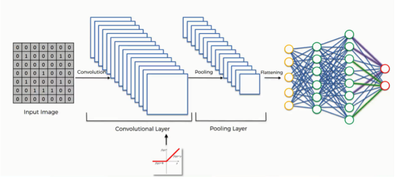


Fig. 6. Convolutional Neural Network

7. Convolutional Neural Network

There are several drawbacks of Neural Networks(NNs). NNs use one perceptron for each input and hence the amount of weights rapidly becomes unmanageable for large datasets. As a result, difficulties arise whilst training and overfitting can occur. Another common problem is that NNs react differently to an input and its shifted version — they are not translation invariant.

CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output. The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply on CNNs. Unlike neural networks, where the input is a vector, here the input is a multi-channeled volume.

The convolution layer is the main building block of a convolutional neural network. The convolution layer comprises of a set of independent filters. All these filters are initialized randomly and become our parameters which will be learned by the network subsequently. Parameter sharing is sharing of weights by all neurons in a particular feature map. Local connectivity is the concept of each neural connected only to a subset of the dataset (unlike a neural network where all the neurons are fully connected). This helps to reduce the number of parameters in the whole system and makes the computation more efficient. A pooling layer is another building block of a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently.

When we tried making our model by making using a convolutional neural network, we found that the model was not good and the accuracy was very low.

8. Hybrid Model

After the dis-satisfactory results of the convolutional neural network, we experimented by creating a hybrid classification model using a Neural Network and a Support Vector Machine. Though the neural network alone was giving good accuracy, we wanted to see if we could increase it. We have previously discussed Neural Networks. We will discuss Support Vector Machines next.

9. Support Vector Machines

Support Vector Machine is a learning algorithm that is responsible for finding the decision boundary to separate different classes and maximize the margin. Margins are the (perpendicular) distances between the line and those dots closest to the line. Obviously, infinite lines exist to separate the points of the dataset. SVM needs to find the optimal line with the constraint of correctly classifying either class.

A hyperplane is an (n minus 1)-dimensional subspace for an n-dimensional space. For a 2-dimension space, its hyperplane will be 1-dimension, which is just a line. For a 3-dimension space, its hyperplane will be 2-dimension, which is a plane that slice the cube. Margin is thus the distance between dataset points and this hyperplane. The point of SVM is thus to find the minimal or optimal distance.

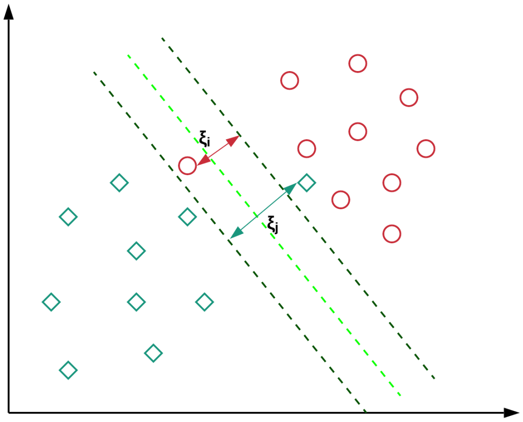
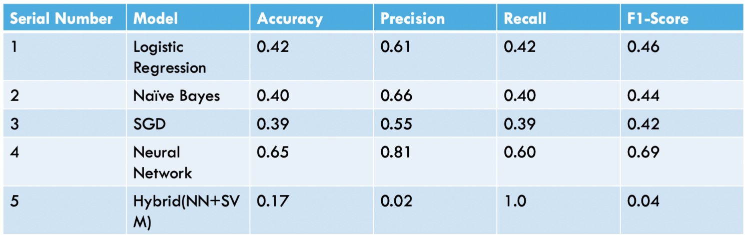


Fig. 7. Support Vector Machine

VIII. Results

After applying each of the discussed models, the following accuracies were obtained.



Serial Number	Model	Accuracy	Precision	Recall	F1-Score
1	Logistic Regression	0.42	0.61	0.42	0.46
2	Naïve Bayes	0.40	0.66	0.40	0.44
3	SGD	0.39	0.55	0.39	0.42
4	Neural Network	0.65	0.81	0.60	0.69
5	Hybrid(NN+SVM)	0.17	0.02	1.0	0.04

Fig. 8. Accuracies of the Various Models Used

From the table we can see that the best model by far is the normal neural network with an accuracy of 0.65. Logistic regression and Naive Bayes classification are both average with around 0.4 accuracy. Our hybrid model was pretty much unsucessful with the lowest accuracy of 0.17
Below is a plot of the accuracies for an easy visualization

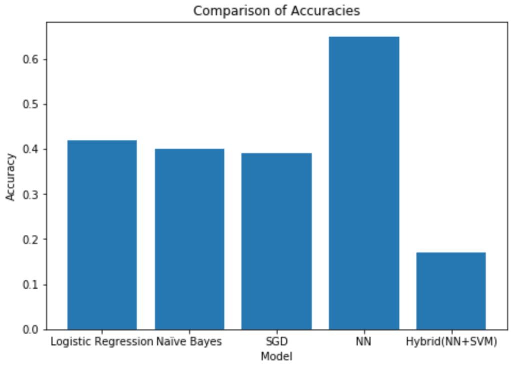


Fig. 9. Accuracies of the Various Models Used

We can plot and compare the recalls of the various models as shown below

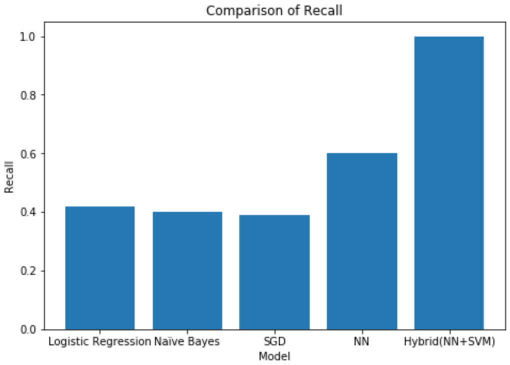


Fig. 10. Recalls of the Various Models Used

The plot and comparison of the precisions of each of the various models is shown below

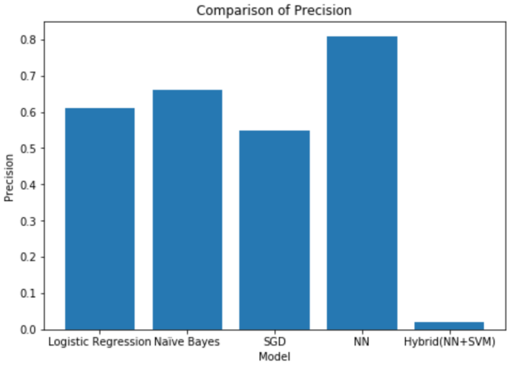


Fig. 11. Precisions of the Various Models Used

The plot and comparison of the F1 score of each of the various models is shown below

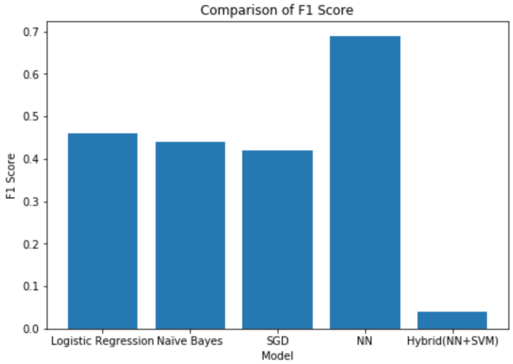


Fig. 12. F1 Scores of the Various Models Used

Shown below are the epoch performance metrics of the Neural Network

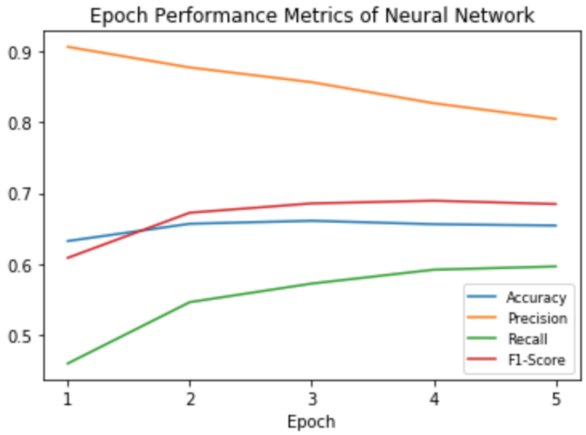


Fig. 13. Epoch Performance Metrics

Shown below are the epoch performance metrics of our Hybrid Network

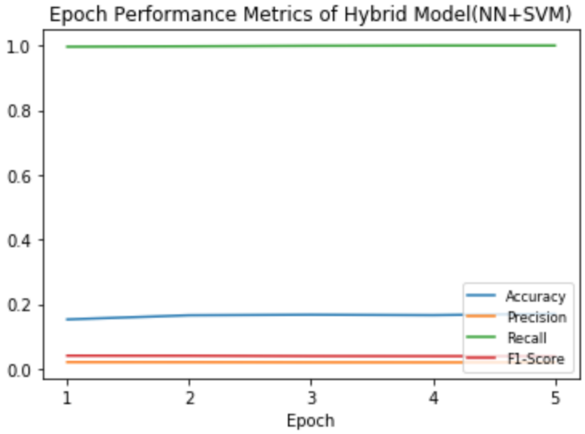
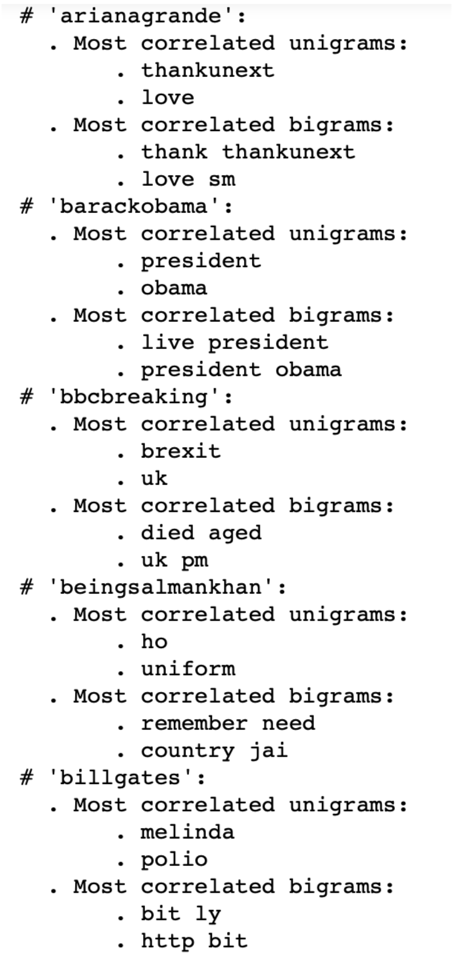


Fig. 14. Epoch Performance Metrics

Finally some the most probable unigrams and bigrams are shown



```
# 'arianagrande':
  . Most correlated unigrams:
       . thankunext
       . love
  . Most correlated bigrams:
       . thank thankunext
       . love sm
# 'barackobama':
  . Most correlated unigrams:
       . president
       . obama
  . Most correlated bigrams:
       . live president
       . president obama
# 'bbcbreaking':
  . Most correlated unigrams:
       . brexit
       . uk
  . Most correlated bigrams:
       . died aged
       . uk pm
# 'beingsalmankhan':
  . Most correlated unigrams:
       . ho
       . uniform
  . Most correlated bigrams:
       . remember need
       . country jai
# 'billgates':
  . Most correlated unigrams:
       . melinda
       . polio
  . Most correlated bigrams:
       . bit ly
       . http bit
```

Fig. 15. Some Most Probable N-Grams

References

[1] Su, Jessica Shukla, Ansh Goel, Sharad Narayanan, Arvind. (2017). De-anonymizing Web Browsing Data with Social Networks. 1261-1269. 10.1145/3038912.3052714.

[2] Shao, Yingxia Liu, Jialin Shi, Shuyang Zhang, Yuemei Cui, Bin. (2019). Fast De-anonymization of Social Networks with Structural Information. Data Science and Engineering. 4. 10.1007/s41019-019-0086-8.

[3] Shirani, Farhad Garg, Siddharth Erkip, Elza. (2018). Optimal Active social Network De-anonymization Using Information Thresholds. 1445-1449. 10.1109/ISIT.2018.8437739.

[4] V. Vincze and R. Farkas, ”De-identification in natural language processing,” 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, 2014, pp. 1300-1303.

[5] Medlock, Ben. (2006). An introduction to NLP-based textual anonymisation.