Email User Classification and Topic Modeling

Krupal Shah[1], Nirav Shah[2] Shaival Shah[3], and Dip Patel[4]

North Carolina State University, Raleigh NC 27695, USA

Abstract. Nowadays, the primary form of communication in most organizations is via email. Thousands of users communicate on daily basis via email. The paper proposes a method to analyze the content of the email and predict the author of the email based on writing patterns of different users using Deep Learning methods. Along with that, this paper also discusses existing methods and their performance for extraction of topics and keywords from the contents of emails.

Keywords: Bidirectional Encoder Representations from Transformers (BERT), Recurrent Neural Networks (RNN), Artificial Neural Networks (ANN), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), Latent Dirichlet Allocation (LDA)

1 INTRODUCTION AND BACKGROUND

1.1 Problem Statement

Email is one of the primary form of communications in an organisation. Each organisation has their own respective email domain. A lot of interesting insights can be obtained from emails of different users in an organisation. But sometimes someone can get access to other email's account and can perform malicious activities.

The primary problem is to analyze the content of the emails and use that to predict the person who wrote that email. We would try to understand the writing patterns of different people and classify emails on the basis of similar writing patterns. Also, we know that this task can be performed using both deep learning and machine learning techniques. We will try to compare the performance of both kind of models and see whether there is a need of computationally complex deep learning model, or we can achieve good results using machine learning models.

We would also try to analyze the contents of the emails and delve deeper into the enron dataset to find out topics of different groups of emails. We will use various existing topic modeling methods to group the emails and extract primary topics of discussion amongst them.

1.2 Related Work

There has been major breakthroughs in the task of text classification over the past decade. Several large and complex deep learning models have proven very effective in the task of text classification. The paper [1] describes the state-of-art approach for extreme multi-label text classification using Bidirectional Encoder Representations from Transformers (BERT). The authors achieved a 11.43% relative improvement over the previous state-of-art model Parabel on the Wiki-500k dataset. We will implement the BERT model for obtaining the BERT vector representation of various email authors. Then we will use those vector representation to identify the authors of email from the content. Also, those vector will be used for the purpose of topic modeling and email clustering.

There are a lot of application of the Enron dataset. The paper [2] describes the Enron dataset in detail and the usefulness of the dataset in email classification. The authors tried to classify emails in the dataset using SVM. They concluded from the results that the Enron dataset is a very valuable dataset for email classification and further research should be carried out on the dataset. The dataset can be used to train models on real life like email data. We will use this dataset as it represents real life like scenarios in email communication. Therefore a model trained on such data will give high accuracy on real emails. We will be using ideas presenting in the paper for the purpose of data preprocessing and email classification.

Clustering algorithms are being developed for finding out similarity among different section of data. The paper [3] describes a new clustering algorithm DBSCAN for the task of class identification in spatial databases. The algorithm relies on the density based notion of clusters for clustering the elements. The algorithm showed good performance on synthetic as well as real data. The algorithm was very effective in identifying clusters of arbitrary shape. We will use DBSCAN algorithm to cluster email vectors based on their topics and recognize the most frequently discussed topics of discussion in the emails.

[h]

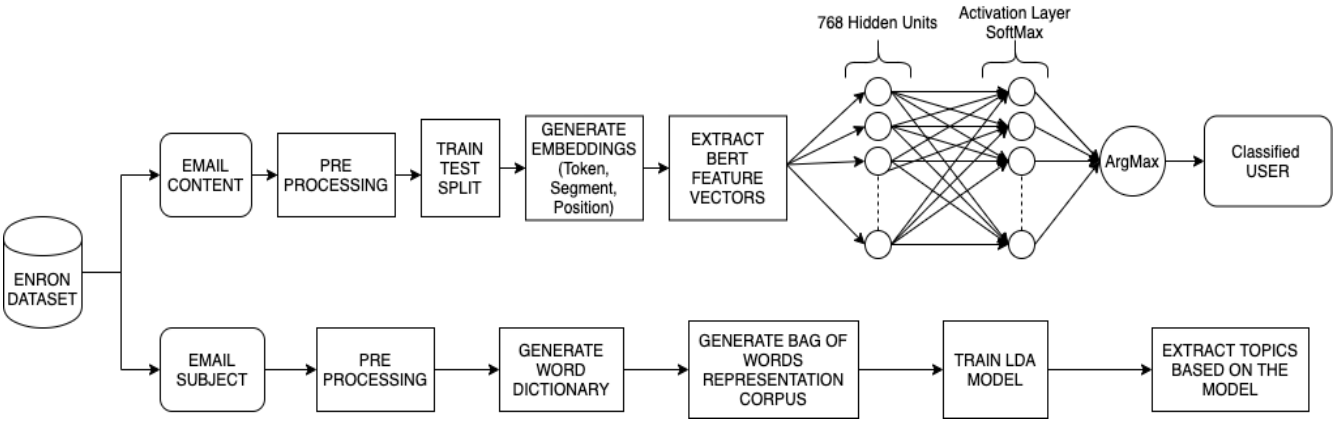

Fig. 1. Final Approach

2 METHOD

2.1 Approach

Proposed Model: Enron dataset has been used for classification of users based on their writing techniques. Also, the dataset has been used to group emails based on similarity and extract frequently discussed topics. Model based on Deep Learning approach for the classification task and Latent Dirichlet allocation (LDA) for extraction of topics is proposed. The detailed architecture of proposed method is given in figure 1.

The first step in the pipeline is to extract the sent email content and subject of that email from the data corpus. This data is then structured and annotated with the respective user. Next in pipeline is to use this data for classification of user emails and extraction of frequently discussed topics.

In order to perform the classification task, the first step is to preprocess the data using various preprocessing techniques such as stopwords removal, stemming,etc. This preprocessing is done mainly to remove information which is not relevant and redundant in the textual data. This data is then split into training, testing and validation subsets.

Classification model has been developed using deep learning approach. For this approach we will be using concepts of NLP to learn the contextual meaning of the text of emails. As of now the state-of-the-art deep learning model is the Bidirectional Encoder Representations from Transformers (BERT) which is a encoder representations of transformer used for advanced NLP tasks. The BERT model is powerful in getting the contextual meaning of the text. BERT's key technical innovation is applying the bidirectional training of Transformer to language modelling. BERT uses combination of three embeddings namely token, segment and position as input.

The next step is to generate those three word embeddings. As shown in the figure 2 input sentence is converted into Token, Segment and Position embedding. Token embedding encodes word vector, Segment embedding is used to differentiate 2 sentences, that is encoding sentences and position embedding encodes the order of words.

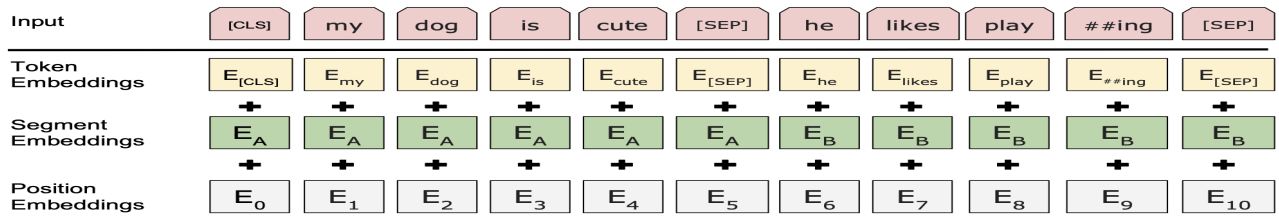

Fig. 2. Input embedding used in the architecture [4]

The embeddings generated are then used by the BERT model for extraction feature vectors. BERT uses an multi-headed attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder

that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder model is necessary. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional (Figure 3). In [5] researchers proposed novel transformer which uses multi-headed attention which surpass results given by

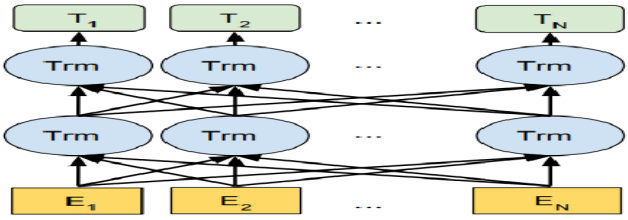

Fig. 3. BERT architecture [4]

traditional RNN and CNN formulas. In figure 3 Trm block is one transformer unit. In [5] transformer architecture consist of encoder and decoder layers where there are 6 encoders and 6 decoders.

Google's pretrained BERT model is trained using two strategies. The first one named Mask Language Model (MLM), which is simple technique to mask some of the words in between the sentence and give to BERT and it tries to predict those masked words. This is how it learns the context of the sentence given to it during training [4]. Other strategy named Next Sentence Prediction, which helps BERT to learn relationship between sentences (The first strategy helps in learning relationship between words in sentence). Here two sentence is given as input and it needs to predict that second sentence given comes next to first one or not. The labels given to predict was IsNextSentence and NotNextSentence [4]. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word). BERT uses combination of preprocessed embedding to output feature vector. This feature vector contains contextual meaning of the emails provided in the input which is then used for the classification task.

The next step in the pipeline for the classification task is to use train an Artificial Neural Network (ANN) network which performs the required classification. Artificial Neural Networks (ANN) are multi-layer fully-connected neural nets. They consist of an input layer, multiple hidden layers, and an output layer. Every node in one layer is connected to every other node in the next layer. Our ANN architecture consists of an hidden layer and an output layer, the hidden layer reads the feature vectors generated by the BERT. The hidden consists of a certain number of hidden nodes. A given node takes the weighted sum of its inputs, and passes it through a non-linear activation function. The first hidden layer represents the mapping between feature vector and hidden units, and the number of hidden units depends on the size of feature vector. The final output layer size depends on the number of users, i.e. number of classes in the classification task. The activation function used in this layer is softmax, since it will map the output between 0 and 1, which can interpreted as the probability that the text belong to a certain class, which is in our case a certain user.

The final step is to read the output of the softmax layer and classify the user based on the probability values.

To extract the frequently discussed topics, Topic modelling using LDA has been implemented. The subject from the sent email corpus of individual user has been extracted. This text is then preprocessed using various preprocessing techniques such as stopwords removal, stemming,etc. This preprocessing is done mainly to remove information which is not relevant and redundant in the textual data.

Word dictionary is generated from the textual corpus of subjects from all emails of all users. Word dictionary is mainly used to represent unique words from all emails and are tagged with integer ids. Using this dictionary each email subject is converted into bag of words representation. The bag-of-words (BOW) model is a representation that turns arbitrary text into fixed-length vectors by counting how many times each word appears. This process is often referred to as vectorization.

This vectorized text is then passed into LDA model for training of the model into distinct topics. In natural language processing, the latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For example, if observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's presence is attributable to one of the document's topics. LDA is an example of a topic model.

Finally top keywords for each topic generated are extracted from the model. These keywords are extracted based on the probability of the keyword occurring in the set of documents and those probability represents how closely that keyword is related to the document cluster in that topic.

Novelty The enron dataset is collection of emails of different users in an organisation. The dataset is available in a highly unstructured manner, and the dataset it cannot readily be applied to any model. We transformed the dataset into an structured format. Extracted the sent emails as it was required for our task and subject of those emails and annotated them with the user to whom the email was sent. This whole process was done automatically via code. Then after preprocessing we used different vector representation method using BERT to convert the text into vector representation so that it can be used with any of the task as required.

The main reason for which the enron dataset was made public was to research and extract the information from the private emails on which they were having conversation. Implemented LDA topic modelling to extract the major topics on which the discussion was going on for e.g. Gas Management. Using these results we were able to find out majority of the topics on which company employees were discussing.

2.2 Rationale

The first step is to preprocess the raw email data which are there in the dataset. We observed that the emails, are available in text raw format, which directly won't be useful in any, machine or deep, learning task. We performed various textual data processing techniques such as removing stopwords, stemming the text in order to remove not so useful information, and finally generate vector representation of the data to feed into the neural network.

We tried to use deep learning architecture for feature extraction and classification. We used BERT as our feature extractor. According to [4] gives state-of-the-art results in getting contextual information from text as compared to previous architectures like LSTM and CNN. Following this paper published by google researchers we decided to use BERT as our feature extractor to extract embedding of the emails. For classification model we used simple naive ANN model. According the [6] they conclude that simple ANN model is sufficient for classification task as higher level features are extracted from BERT model.

For topic modelling we tried to use DBSCAN and LDA and got better results with LDA. The main reason for selecting DBSCAN algorithm was that it is a density based clustering algorithm, using this we tried to find highly correlated emails and cluster them into single topic. As emails would be highly correlated they would be tightly together on the metric plane(i.e. cosine) and DBSCAN would lend better results but thats not what happened and we got biased clusters so we used LDA. LDA is the main approach for topic modelling it was designed to be used on textual data mainly documents and cluster them into distinct topics based on the similarity of the document with the other documents in independent clusters. LDA gave use very good results for e.g. one of the topic keywords that were extracted were "agreement","gas","letter" from which we could infer that the employees were talking about the gas management and this might actually be true as enron was an oil and gas corporation. So we decided to use LDA for finding topics amongst the emails.

3 EXPERIMENTATION

3.1 Dataset

The dataset used for this project is ENRON dataset [7]. This dataset was collected and prepared by the CALO Project (A Cognitive Assistant that Learns and Organizes). It contains data from about 154 users, mostly senior management of Enron, organized into folders. The corpus contains a total of about 0.5M messages. The dataset had 154 folders, one for each user, and each folder had sub folders 'sent items', 'inbox', 'attachments', 'spam', etc containing respective emails and other things in different formats The enron dataset is available in raw email format which needs to be preprocessed for applying Natural Language Processing techniques to learn from the data. We have used various preprocessing techniques to filter out useful data and created a dataframe which is then used for relevant tasks.

Starting from filtering out only sent mails for individual users to be trained on deep learning and machine learning model. As these mails are raw format emails from gmail, they do contain email threads and metadata which need to be filtered out. We use regular expression to filter out as well as some search and eliminate techniques automatically to remove unuseful snippets of email. Also some of the emails contain threads from other users which were also eliminated. Only emails from the user sent were considered, other thread emails were ignored and each of these emails were tagged with the individual user name.

After this tagging we pass each of this text through basic preprocessing which removes extra lines from the text, removes stop-words such as "and","the","in",etc. We also filtered out the URL snippets such as "https" from the text. All the text is converted to lower text and this text contains all useful words separated by space.

To make this information more useful and to embed this words into word embedding in proper sense we also stem the words. Stemming is a method in which we try to convert each word into its root word using various stemming techniques. For e.g. we stem word "singing" into "sing" and so on. We have used a pre-implemented stemmer from nltk named snowball stemmer for stemming the text from input. Each of the word for each email has been stemmed and joined again so that we can learn maximum information from this text. This is the last stage of preprocessing. After this the text is used for email user classification and clustering emails as well. The following sequence shows an example of how the data is being processed at individual stage as described above.

1. **Original Text**: Thanks for sending me email.¡a href=img.png¿
2. **Removing URL**: Thanks for sending me email.
3. **Removing Stopwords**: Thanks sending email.
4. **Removing Punctuation**: Thanks sending email
5. **Lower Case**: thanks sending email
6. **Stemming**: thank send email

The size of the final dataframe created, using the above described procedure and, is 12628*2. It has two column, the first column has the email text data sent by a user and the second column has the respective user name.

3.2 Hypothesis

The main goal of this project is to investigate whether we can identify which user has written an email. This can useful in case of an intrusion, we can identify whether a specific employee has sent an email or some intruder has gained access of that employee's email and sent. Also, we can identify the credibility of an email using the model. The other goal is to perform email clustering via analysing email content and group emails into various topics. We can use this to find out list of hot topics around which majority of company emails are aligned. Lastly, we can also analyze whether deep learning models, which are more complex and hard to train, are really necessary for classification or we can get good results using machine learning models only.

3.3 Experimental Design

In order to evaluate our first hypothesis, i.e. investigate whether we can identify the user of an email along with our third hypothesis, i.e. which is to analyze and compare the performance's of deep learning and machine learning model , we created and experimented with two classification model one based on machine learning and other based on deep learning. The task of each classification model was to predict the author of the email based on the email text. If the author predicted by the model matches the sender then we can say that the email is not fraudulent and vice versa. For experimentation purpose, initially all the classification models were trained and tested on subset of data which consists of 15 user and their respective emails. Based on performance of each model, the final selected model was trained and tested on dataset consisting of all users. There were total of 154 users considered for evaluating the performance for the final model. The final baseline accuracy for the classification task is based on the paper [2] which is around 69%. Both, machine learning and deep learning, models follow the same general pipeline, the design of which is shown in Fig 4. The details about each developed models are given below:

Machine Learning Model The machine learning model is developed for the purpose of user email classification. The first step in the pipeline is to preprocess the raw enron data in order to generate dataframe which can be used for the purpose of classification. The entire process is given in detail in section 3.1. After creating the dataset, we splitted the dataset into training and testing dataset with the split ratio as (80,20). The dataset comprised of textual data, and we cannot feed raw text data to a machine learning model. In order to feed data to the model, we have implemented and experimented with two preprocessing techniques. The techniques we used to convert the textual data to vector form are TF-IDF and CountVectorizer.

CountVectorizer will transform text data into its vector representation, which is similar to one-hot encoding. It converts a collection of text documents to a matrix of token counts. Here, each text is represented in vector form, and each word in the text is replaced with 1 if the word is there in the text or 0 if the word is not present. Similarly, TF-IDF short for term frequency–inverse document frequency, is a numerical statistic which reflects how important a word is to a document in a collection or corpus. The tf–idf value increases proportionally to the number of times

a word appears in the document and is offset by the number of documents in the corpus that contain the word. The final output will be a vector where each word in an entry is represented by its, TF-IDF score of that word with respect to all the other words in case of TF-IDF processing method, 0s and 1s depending on whether the word is present or not in case CountVectorizer processing method.

The next step is to perform feature selection,Feature selection is the process of selecting a subset of relevant features for use in model construction. We performed model feature selection using embedded methods. Embedded methods use algorithms that have built-in feature selection methods. We performed feature selection using LogisticRegression model with L2 regularization. L2 regularization adds the L2 norm penalty to the loss function. The coefficients are squared in the penalty expression, it forces the coefficient values to be spread out more equally, and correlated features tend to get similar coefficients. The output of this process, is subset of relevant features which is then used for model construction.

Next step is to train the machine learning model. We experimented with 3 machine learning models which are: LinearSVC, SGDClassifier and RandomForest. These specific models were chosen based after researching about the performances of each on various tasks related to textual data.

LinearSVC is based on Support-vector machines, which are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Support Vector Machine (SVM) uses support vectors (data points near the decision boundary) to maximize the margine between 2 classes. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. As our problem is of multiclass classification we used multiclass SVM which gets n-1 number of boundaries given total classes are equal to n. [8]

Stochastic Gradient descent classifier is a regression approach which uses stochastic gradient descent to minimize the loss and reach the optimal point. This estimator implements regularized linear models with stochastic gradient descent (SGD) learning: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule (aka learning rate). [9]

Random Forest Classifier is an ensemble tree-based learning algorithm. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. It aggregates the votes from different decision trees to decide the final class of the test object. [10]

During training of each model, we performed 3-fold cross validation, and the cross validation accuracy is calculated as the mean accuracy of all 3 folds. We trained each ML model with its default configuration, and selected the best model based highest classification accuracy. Once the final model is selected performed hyper parameter tuning for the final model using GridSearchCV in order to select the best hyperparameters which yields the highest classification accuracy. Experimentation has been done on hyperparameters given below for the best model:

- C: Regularization Parameter
- max_iter: Maximum number of iteration
- Penalty: Norm used in Penalization

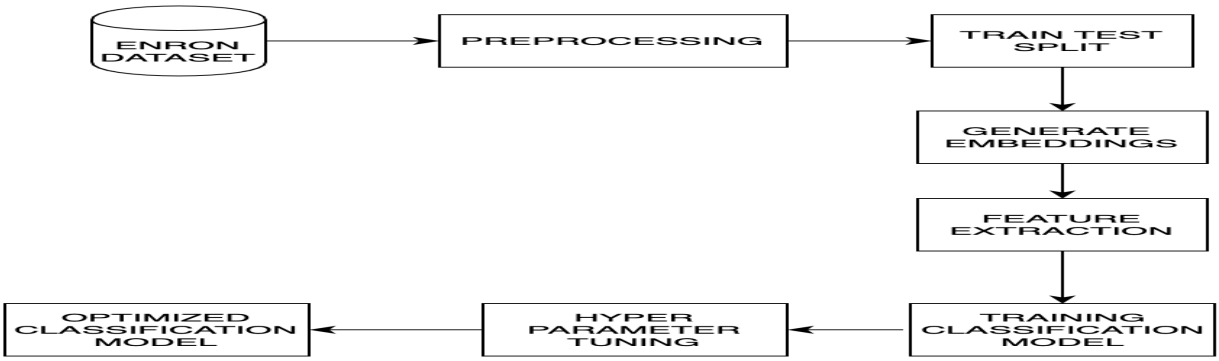

Fig. 4. General experimentation pipeline

Deep Learning Developed and implemented two classification models, one based on BERT architecture and other based on Long Short Term Memory (LSTM).

For the BERT model, similar to machine learning model design deep learning model also follows our general experimentation pipeline. Here we followed same approach as machine learning till splitting the emails dataset into training and testing set. Instead of using TF-IDF vectors BERT uses token, segment and position encodings and we used pretrained google's BERT which generates these embedding and extract features. For classification, ANN model was used to classify emails. In our ANN model, Adam optimizer was used with initial learning rate 1e-2 with categorical cross entropy loss function as this is multi class classification model and classification accuracy metric is used to evaluate and compare models. The ANN model consists of one hidden and one softmax output layer. The first hidden layer represents the mapping between feature vector and hidden units, and the number of hidden units are 768, as BERT model outputs vector of 768 dimensions. The final output layer has 15 neurons, as we are performing classification of 15 users during experimentation. The activation function used in this layer is softmax, since it will map the output between 0 and 1, which can interpreted as the probability that the text belong to a certain class, which is in our case a certain user. Used pretrained google's BERT model and only learned the parameters of classification model. Experimentation was done on hyperparameters batch size, learning rate, epochs and maximum sequence length.

For the LSTM model, we followed same pipeline as our BERT model. First we did data preprocessing. Here the word embedding are extracted using google's Word2vec model.

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space.

The Word2vec model extracted embeddings of maximum sequence length of 70 and 100 dimension vector. Similar to machine learning model the dataset was spillted into training and testing with testing split of 20% and for performing validation during training we splitted the training set into trainig and validation with split of 20%.

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture. LSTM has feedback connections, it can not only process single data points, but also entire sequences of data such as text sequences or video.

In our model, LSTM acts as a feature extractor of input sequence of email text and fully connected layer is used for classification. The LSTM model consisted of two LSTM layers with 128 units each and dropout of 0.3, 3 hidden layers with 64 units each and one softmax layer. The number of units in the softmax layer depends on the number of classes in the classification, and is used at the end to output probabilities of all classes. The model was trained using Adam optimizer with categorical cross entropy loss, learning rate of 0.01, and accuracy as our metric of measure. The below graphs represents the change in model accuracy as the model trains.

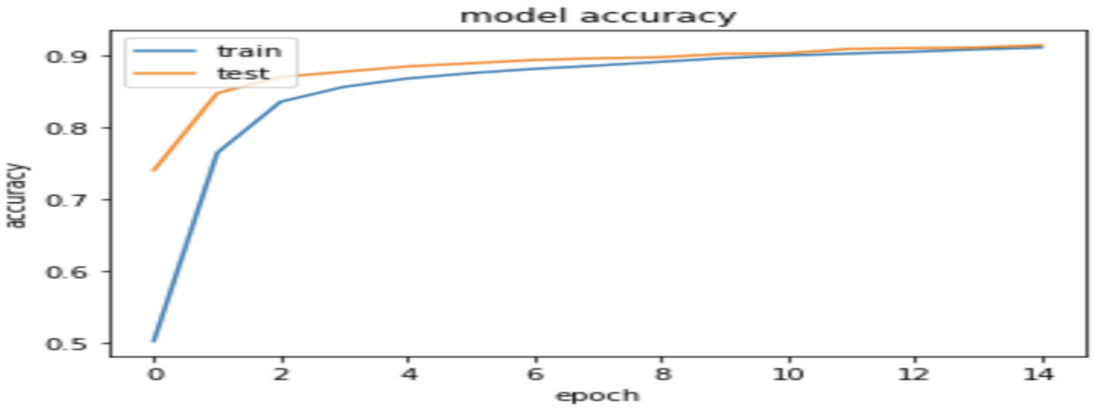

Fig. 5. Training accuracy curve

In order to evaluate our second hypothesis, whether we can find the keywords or topic from a set of emails and whether we can cluster similar emails we performed topic modeling using DBSCAN and LDA. The details about this is given below:

Topic Modelling To achieve topic modelling and to extract relevant high frequent topics amongst the emails we used DBSCAN initially but due to lack of good results LDA was implemented which resulted into better extraction of topics.

For our dataset we tried two approaches to feed data into DBSCAN algorithm. We converted the sent emails of the users into vectors using doc2vec and tagged them using the user to whom the mail was sent. We trained the doc2vec model for 10 epoches to train on our data and generate vectorize representation. Then using that trained model each and every email which was preprocessed was converted into vectors. These vectors were passed into the DBSCAN algorithm with eps=0.11 and minimum samples=3. We used cosine similarity for this so that the vectors can be compared appropriately in each dimension. The eps was selected based on the 2-NN frequency graph which took sharp increase after 0.11. It is shown in the figure below

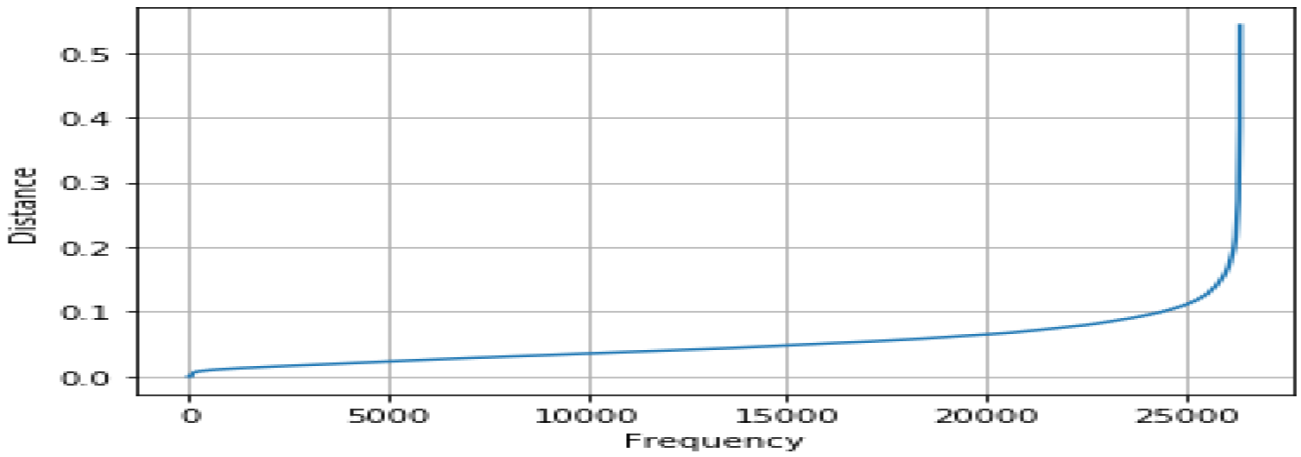

Fig. 6. Elbow Plot

The problem with this approach was that despite of tuning the parameters biased clusters were generated. i.e. Almost 80 Percentage of emails were getting collected in single cluster and rest being distributed among other clusters.Another method to represent these emails was using the BERT generated vectors from the Bert model we used for classification. For this representation as well we were getting the same problem. The majority of the emails were clustered into a single cluster.

The final approach which was implemented was using Latent Dirichlet allocation. Also instead of using the email content, the subject of the emails were used so that proper topic identification can be done. For this approach the email subject were extracted from each email and then preprocessed. These preprocessed subjects were converted into bag of words representation. A common dictionary corpus out of the input data was generated for inverse mapping of words. These two things were then fed into Latent Dirichlet allocation algorithm to generate different topics. LDA also clusters the documents based on their relation with the current cluster and the cluster in which it will be transferred. If resultant cluster has more similarity then the document is transferred to that cluster. This process is iterated several times, In this case these process was iterated 20 times and all the docs were clustered into 15 different topics. The reason for selecting 15 different topics was based on trial and error method, for number of topics above this we were getting some redundant keywords in different topics and we wanted maximum topics to be generated so 15 was selected as number of topics. Finally based on top keywords we were able to find out the topic keywords for each topic as shown in the results section.

4 RESULTS

4.1 Classification Models

Classification Model using Machine Learning: Implemented various classification model, using LinearSVM, SGDClassifier and Random Forest algorithms, and two text processing methods TF-IDF and CountVectorize.

Evaluated the performance of each model using classification accuracy achieved on the test set.
Using TF-IDF and ML Algorithms: Implemented TF-IDF vectorization method for representing preprocessed text. The results are as follows:

Algorithm	**CV Accuracy**	**Test Accuracy**
LinearSVC	86.95%	87.74%
SGDClassifier	83.83%	84.07%
RandomForest	84.13%	85.07%

We can observe that each classifier has similar accuracy, Linear Support Vector Classifier is the one with the highest testing accuracy.

Using CountVectorizer and ML Algorithms: Implemented CountVectorizer vectorization method for representing preprocessed text. The results are as follows:

Algorithm	**CV Accuracy**	**Test Accuracy**
LinearSVC	80.7%	80.1%
SGDClassifier	71.73%	70.61%
RandomForest	78.95%	79.3%

Based on the test accuracy of each machine learning model with respective vectorization method, we can conclude that Linear Support Vector Classifier along with TF-IDF vector representations was able to correctly classify emails of different users with the highest accuracy as compared to other models. Hence, we performed hyper parameter tuning using GridSearchCV with 5-fold cross validation on the model to see whether we can increase the performance of the model. The hyper parameters chosen for tuning along with their respective range of values are as follows:

- C: 0.1,0.3,0.5,1
- penalty: l2,l1
- max_iter: 500,1000,2000,3000

The top 3 results, sorted based on accuracy, are as follows:

C	**max_iter**	**Penalty**	**CV Accuracy**
1	500	l2	88.03%
0.5	500	l2	87.91%
0.3	500	l2	87.56%

The LinearSVC model with the best result with final accuracy on the test set is 88.71%.

Classification Model using Deep Learning: Implemented two classification model, using BERT and LSTM models, and evaluated the performance of each model using classification accuracy achieved on the test set.

LSTM Model: Implemented LSTM model using the following parameters:

- Number of Hidden Layers in LSTM: 2
- Number of FC Layers: 1
- Number of neurons in LSTM: 128
- Number of neurons in FC Layers: 64
- Batch Size: 32
- Learning Rate: 0.01
- Number of Epochs: 15
- Max_Seq Length: 70

The results are as follows:

Model	**Validation Accuracy**	**Test Accuracy**
LSTM	92.26%	90.72%

BERT Model: Implemented BERT model using the following parameters:

- Train Batch Size: 16
- Eval Batch Size: 16
- Learning Rate: 0.00001
- Number of Epochs: 1
- Max_Seq Length: 100

The results are as follows:

Model	**Test Accuracy**
BERT	92.21%

Based on the test accuracy of both models, we can conclude that BERT was able to correctly classify emails of different users with the highest accuracy as compared to LSTM model. Hence, we performed hyper parameter tuning on the BERT model to see whether we can increase the performance of the model. The hyper parameters along with their respective results are as follows:

Train Batch Size	**Eval Batch Size**	**Learning Rate**	**Epochs**	**Max_Seq**	**Accuracy**
32	32	1e-5	5	50	95.32%
64	16	1e-2	5	100	95.73%
32	8	1e-5	3	50	95.58%
128	16	1e-3	10	150	93.84%
32	16	1e-5	5	50	96.46%

The final accuracy we got on the test set using the optimized BERT model is 96.46%.

On comparing both, machine learning and deep learning models, we can observe that BERT model has the highest accuracy. Thus, our final classification is based on BERT. On training and testing the optimized BERT model for all 154 user's email, the final accuracy achieved is 82.28%.

4.2 Topic Modelling Results

For topic modelling we tried to implement DBSCAN clustering algorithm on Doc2Vec vectors of the Emails and Bert Output vectors of the Emails. For both of those the result was the biased clusters as shown below Final Topic

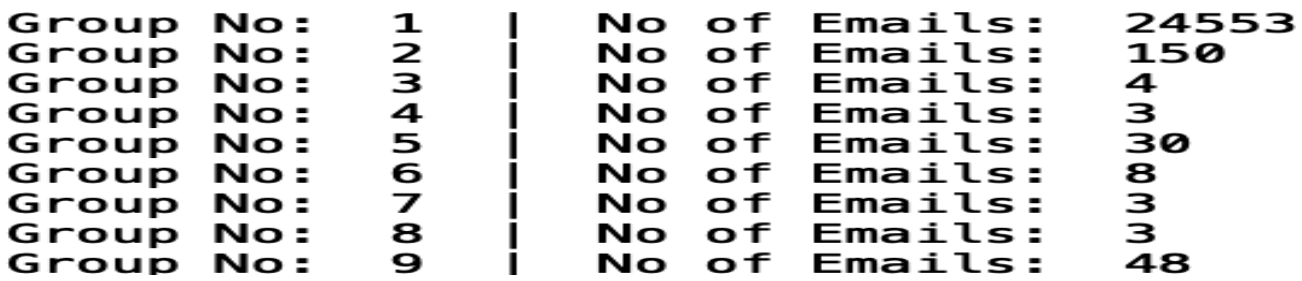

Fig. 7. DBSCAN Results

modelling was done using LDA and using 15 topics and 3 top keywords. The result of this approach is shown in figure below. It also shows the probability distribution of the top keywords to give rough idea of the presence of those keywords in the Topic groups. The left most number in the individual tuples represent the topic number and the right most part with probabilities multiplied indicates the most frequent keywords. For e.g. "gas","agreement","letter" indicates that main discussion topic might be based on a agreement letter based on gas and this might be true as enron was an oil and gas corporation.

```
(0, '0.087*"agreemen" + 0.083*"gas" + 0.054*"letter"')
(1, '0.168*"meeting" + 0.113*"good" + 0.101*"deals"')
(2, '0.122*"deal" + 0.066*"man" + 0.042*"presentation"')
(3, '0.186*"request" + 0.052*"houston" + 0.039*"gallup"')
(4, '0.360*"fwd" + 0.063*"subject" + 0.062*"prayer"')
(5, '0.075*"ena" + 0.057*"party" + 0.056*"purchase"')
(6, '0.095*"reques" + 0.057*"today" + 0.057*"jan"')
(7, '0.097*"eol" + 0.074*"project" + 0.071*"hey"')
(8, '0.068*"price" + 0.044*"business" + 0.038*"plan"')
(9, '0.065*"approval" + 0.047*"may" + 0.037*"transwestern"')
(10, '0.076*"revised" + 0.062*"story" + 0.049*"documents"')
(11, '0.228*"subject" + 0.069*"energy" + 0.048*"children"')
(12, '0.152*"new" + 0.097*"daily" + 0.083*"call"')
(13, '0.207*"enron" + 0.048*"upda" + 0.045*"review"')
(14, '0.064*"contract" + 0.051*"announcemen" + 0.048*"please"')
```

Fig. 8. LDA Results

```
Sender:  neal-s
Subject:   Re: Need your approval - Generic ID Request -- 2nd Letter
Content:
Bhavna Pandya
08/09/2000 09:59 AM
To: Scott Neal/HOU/ECT@ECT
cc: Roberto Deleon/Corp/Enron@Enron
Subject: Need your approval - Generic ID Request -- 2nd Letter

Sender:  fossum-d
Subject:   TW/El Paso
Content:  No problem with the plan to send a demand letter to El Paso on the
measurement issue.  Please include me on distribution for the draft.
Thanks,  DF

Sender:  gay-r
Subject:   Fwd: GE Letter Agreemen
Content:
As requested, attached please find the e-mail I sent to Rob Gay to which the
current draft of the Letter Agreement with GEII is attached and which briefly
summarizes the indemnity issue we have been discussing.
```

Fig. 9. Topic Emails

We can verify the validity of these topics from some of the emails as shown below As seen in the figure above we can conclude that all the above emails were closely related to each other as the discussion was done on some of the legal matters which involved letter or agreement in their discussion. This is just a subset of the emails clustered into this topic.

4.3 Discussion

From the above results we can discuss on following aspects,

Performance of Classification Model The hypothesis, that whether we can classify certain email, was supported and the model was able to classify emails of different users. The paper [2] also performed classification of emails based on user's writing skills achieved 69% using SVM Model. Our proposed model was able to beat that accuracy and achieved 82.28% accuracy.

Also, deep learning model were able to outperform than machine learning model because machine learning models require vector representation of text data in order to perform classification. Methods like CountVectorizer and TF-IDF performs such transformations but they don't capture sequential and semantic meaning of the text data, and it is important in order to learn the nature of the sentence.

Using Bert Feature Vectors BERT extracted features gave the best results comparing with other feature extractor used in the project. Based on the results, we can say that BERT extracted feature gave better contextual meaning than all other feature extractors in our project. Due to the bidirectional sequence reading, multi-headed self attention layers and layer normalization used in BERT model, it can extract text specific features using which classification model can do prediction task with highest accuracy.

Using Latent Dirichlet allocation For topic modelling and extracting frequently discussed topics LDA was used which gave relevant results as compared to DBScan clustering. As shown in the results section semantically related

emails were clustered into single topic and the topic keywords extracted also summarized the whole cluster. These results were promising and were as suggested in the hypothesis.

5 CONCLUSION

Following things can be concluded from the above results and approach used:

- Enron dataset has abundant amount of data from which many things can be inferred and extracted. The preprocessing technique used for different dataset depends on the type of dataset and its structure. The classification of users and the topic extraction were just a glimpse of techniques which can be applied to this dataset.
- It was concluded from the above experimentation that BERT model outperformed all other models. From this we can learn that, when dealing with textual data, the performance of a model for any task, depends heavily on how textual data is processed and the information represented by the feature vectors. Specifically, the feature vectors should able to capture the semantic and sequential meaning of the textual data for better performance.
- Also the failed experimentation gave us some knowledge about the nature of the dataset. DBScan algorithm was not able to give proper clusters because the data might have/had many email blobs based on the vector representation we had. This shows that though DBScan is generally good for density based clustering but might not be appropriate to use for certain kind of dataset such as the enron dataset.
- Latent Dirichlet Allocation gave promising results as per the requirement and the topics extracted based on the parameters used were quite summarizing for the cluster of emails in corpus. The technique was applied and tested on the subject of the sent emails but can also be tried with some other content of the dataset.

6 FUTURE WORK

There were only limited numbers of parameters for the dataset considered for the classification model. More experimentation could be done with more information about the different attributes, such as demographics of a user which can be an important factor, for the classification task in order to analyze and improve the performance of the model. Experimentation regarding topic modeling has shown informative results but there were some backlogs in some cases, which implies that more research and experimentation is needed in this part in order to extract significant results, i.e. topics of keywords, out of email text.

References

1. W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. Dhillon, "X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers," 2019.
2. B. Klimt and Y. Yang, "The enron corpus: A new dataset for email classification research," in *European Conference on Machine Learning*. Springer, 2004, pp. 217–226.
3. M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise."
4. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.
5. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.
6. M.-W. C. Jacob Devlin, "Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing," 2018.
7. V. Vanburen, D. Villarreal, T. McMillen, and A. Minnicks, "Enron dataset research: E-mail relevance classification," *Technical Reports-Computer Science*, 01 2009.
8. T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.
9. S. Diab, "Optimizing stochastic gradient descent in text classification based on fine-tuning hyper-parameters approach. A case study on automatic classification of global terrorist attacks," *CoRR*, vol. abs/1902.06542, 2019. [Online]. Available: http://arxiv.org/abs/1902.06542
10. B. Xu, X. Guo, Y. Ye, and J. Cheng, "An improved random forest classifier for text categorization," *Journal of Computers*, vol. 7, 12 2012.