SIGNATURE EXTRACTION FROM E-MAILS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Divya Muttineni

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

October 2016

Fargo, North Dakota

North Dakota State University
Graduate School

Title

SIGNATURE EXTRACTION FROM E-MAILS

By

Divya Muttineni

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Juan Li
Chair

Dr. Vasant Ubhaya

Dr. Na Gong

Approved:

11/17/2016	Dr. Brian M. Slator
Date	Department Chair

ABSTRACT

Detecting user identity information from the email is one of the predominant exploring topics in data mining. One approach is to extract signature from the body of emails. Those names are usually suitable for representing the sender's or recipient's identity. To overcome the limitation, we proposed the novel approach to extract the signature of email sender and recipient from salutation and signature blocks and email bodies. After locating and extracting signature blocks from email bodies, we can identify the names in the salutation and signature lines, which can be directly associated with the corresponding email address in email headers and the body by using named entity recognition (NER) tools. For these tasks Naive Bayes, maximum entropy and Support Vector Machines (SVM) algorithms are preferred in this paper. Results on the data subset of the Enron corpus indicate that the approaches presented in this paper can extract signature from email bodies.

ACKNOWLEDGEMENTS

Though only my name appears on the cover of this disquisition, there were many people who have contributed to its production. I owe my gratitude to all those people who have made this possible and because of whom my graduate experience has been one that I will cherish forever. I would like to extend my deepest gratitude to my advisor Dr. Juan Li for introducing me with this topic and giving me an opportunity to work under her guidance. I am very grateful to my committee members Dr. Vasant Ubhaya and Dr. Na Gong for their valuable time and suggestions regarding my Paper. Finally, I would like to thank my parents and my friends, who have supported me throughout entire process, by helping me putting pieces together. I will be grateful forever for your support.

TABLE OF CONTENTS

ABSTRACT iii

ACKNOWLEDGEMENTS iv

LIST OF FIGURES vii

CHAPTER 1. INTRODUCTION 1

1.1. Signature Formulation in Email 2

1.2. Signature Extraction in Email 3

CHAPTER 2. BACKGROUND AND RELATED WORK 4

2.1. Data Mining 4

2.2. Natural Language Processing 5

2.2.1. Named Entity Recognition 5

2.2.2. Classifier 6

2.3. Related Work 7

CHAPTER 3. METHODOLOGY 9

3.1. Outline 9

3.2. Preprocessing 10

3.2.1. Header and Body Extraction 10

3.3. Signature Extraction 10

3.3.1. Design of Signature Extraction System 11

3.3.2. Extracting Signature 13

3.3.2.1. Name Boundary 13

3.3.2.2. Tokenization 15

3.4. Hash Technique Mechanism..16
3.5. Training Classifiers..16
3.5.1. Naïve Bayes Classifier..18
3.5.2. Maximum Entropy Classifier 20
3.5.3. Support Vector Machine..21
3.6. Implementation 23
3.6.1. Dataset Loader..23
3.6.2. Extraction Phase 24
3.6.3. Result Phase..25
CHAPTER 4. EVALUATION 26
4.1. Experimental Design...26
4.1.1. Dataset 26
4.1.2. Evaluation Metrics..26
4.1.2.1. Precision..26
4.1.2.2. Recall 26
4.1.2.3. F1 Measure..27
4.2. Experimental Results 27
4.2.1. Comparison..27
CHAPTER 5. CONCLUSION AND FUTURE WORK 31
5.1. Conclusion 31
5.2. Future Work...31
REFERENCES 32

LIST OF FIGURES

Figure Page

1: Format of a Signature 3

2: Email Format 9

3: Framework of Email Signature Extraction System 11

4: Flow of Signature Extraction System 13

5: SVM Boundary Segregation 22

6: Dataset Loader 23

7: Signature Extraction 25

8: Result Set 25

9: Number of Datasets Vs Accuracy 28

10: Number of Datasets Vs F1 28

11: Number of Datasets Vs Precision 29

12: Number of Datasets Vs Recall 29

CHAPTER 1. INTRODUCTION

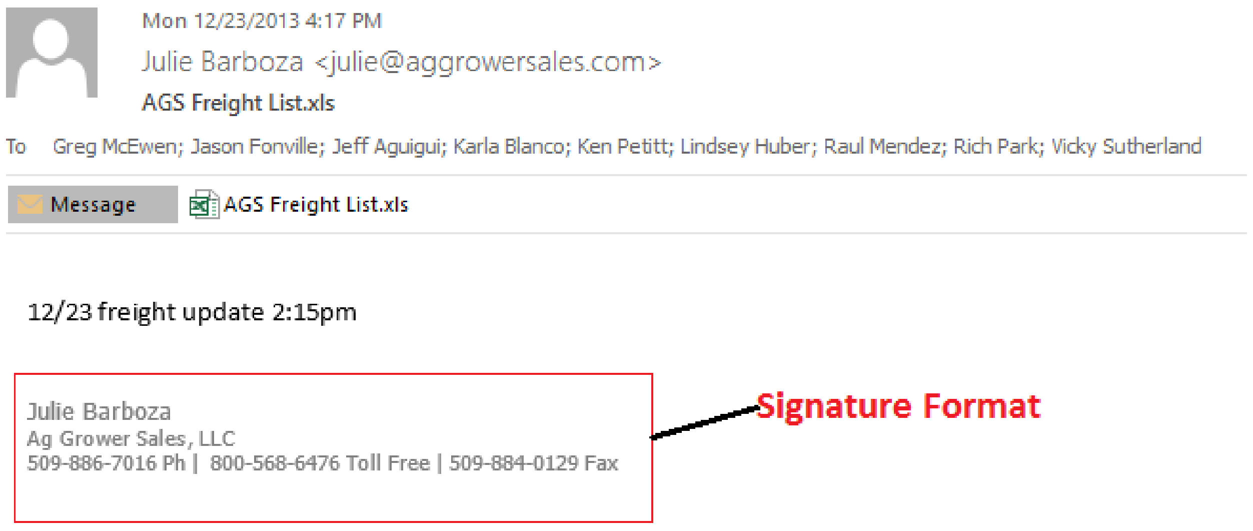
From the early 1950's history of Internet began with the development of computers. Since then the popularity of the Internet makes people communicate in many ways such as Email, Blog and MSN [30]. Among them Email is one of the most used communication service. Email is the concept of sending the messages to an individual person or between the parties. When we usually send emails, people use signatures when communicating with each other. However, the signatures may be different depending on the individuals or based on the company information. Signature is nothing but the One's identity information which may contain person's name, organization, location, phone and so on. Now the signature extraction in emails has become an important research topic of data mining. Data mining can be used in many network applications, such as identity recognition, information retrieval, social network analysis and so on [9]. This paper focuses on the problem of extracting signature from email corpus.

Most approaches extract a user's name from header fields of an email, such as "To" and "From" Header by some default methods, but extracting the signature form the body is the challenging task as we need to identify the person's identity from the signature block. In email bodies, signature usually appear in the last 10 lines of an email message where the first line can be the user name which can be directly associated with the corresponding email addresses in email headers. The first step for signature extraction is to locate and elicit salutation blocks from email bodies. And later we can extract the signature lines. To effectively extract signature lines from the body of an email, we define some features based on the Natural Language Processing mechanisms and later we propose an approach where we use three algorithms to evaluate our results which is presented in our former work.

In this approach we are developing a signature extraction tool that uses three supervised methods namely Naïve Bayes, Maximum Entropy and Support Vector Machine (SVM) to extract the signature from the emails. First we take the email corpus, from that we try to locate and elicit salutation and signature lines from email bodies by using Named entity recognition (NER) or part-of-speech tagging tools to identify the signature lines. Using name boundary word template built on the characteristics of alias neighboring words, which only come with the user names appearing salutation and signature lines, we verify and amend the potential signature that were identified by NER tools. Results on the public subset of the Enron collection indicate that the approaches presented in this paper can efficiently extract signature from the body of emails.

1.1. Signature Formulation in Email

A signature is a set of sequential lines in an email, which identifies the sender's personal information, such as person's name, job title, organization, location, contact information, and the relationship to other important entities. The signature often begins with the sender's name or nickname. If we denote an email E = {l i} i=1 N, where l i is the ith line in E, its signature is S = {l j} j=m n, $1<m<m \ll N$ [1].


Mon 12/23/2013 4:17 PM

Julie Barboza <julie@aggrowersales.com>

AGS Freight List.xls

To Greg McEwen; Jason Fonville; Jeff Aguigui; Karla Blanco; Ken Petitt; Lindsey Huber; Raul Mendez; Rich Park; Vicky Sutherland

12/23 freight update 2:15pm

Julie Barboza
Ag Grower Sales, LLC
509-886-7016 Ph | 800-568-6476 Toll Free | 509-884-0129 Fax

Fig 1: Format of a Signature

1.2. Signature Extraction in Email

For the task of extracting signature block lines, we represented each email document not as a set of features, but as a sequence of lines. Each line is labelled as to whether or not it is part of a signature block, and represented as a set of features. This approach thus reduces the signature line extraction problem to a sequential classification problem. Sequential classification problems can be addressed with MAPI Message Parser class which is used to load .msg files. We also observed improvements on this task by representing lines not only with its own features, but also with the features derived from neighboring lines.

The remainder of this paper is organized as follows. Section II reviews background and related work. Algorithm methodology of system framework to extract signature blocks in email bodies are introduced in section III. The method is evaluated on the data subset and experimental results are shown in section IV. Results of our approach are concluded and future works are discussed in section V.

CHAPTER 2. BACKGROUND AND RELATED WORK

2.1. Data Mining

There is numerous amount of data that is available everywhere. To organize useful data from such a vast source takes lot of time and effort which should not be the case. So, for this we follow a technique called Data Mining. Data mining is the process of extracting non trivial data from huge datasets. Data mining is assisting various applications for required data analysis. Recently data mining has become an important component in extracting information because of the availability of large data. There are two types of datamining techniques which are used. One is the descriptive one which gives information about the existing data and the other is predictive datamining which predicts based on the data [35]. And there are different data mining approaches like classification, clustering, association rule, and outlier detection which are frequently used to analyze network data to gain information extraction related knowledge. The process of data mining includes many steps, which starts with preparation of data from various sources and ends with the presentation of the data mining results. In addition, it is accepted that data mining is not a "one shot" process, but rather it follows the iterative way and continues till it gets the refined or accurate result [34]. In this project, our goal is to automatically extract signature from using data mining methods that accurately detect signature from Enron corpus of email. Generally, Data mining methods detect patterns from large amounts of data, and uses these patterns to detect future instances in similar data [6]. Here in our paper we are using three classification techniques to classify the signature from the email corpus. A classifier is nothing but a rule set, or detection model, which is generated by the data mining algorithm and it is trained over our input data which is otherwise called the training data.

2.2. Natural Language Processing

In the early 1950's, Natural Language Processing began as the intersection of artificial language and linguistics [2]. Natural Language Processing (NLP) is the component of Artificial Intelligence which has the ability to understand the human speech. With the help of Natural Language processing we can be able to communicate with the computer easily like the humans communicate with each other. The great challenge that lies in the text extraction is in the development of automatic methods for the management of text, rather than just its transmission, storage, or display. There were many automatic methods, for indexing large document collections and classifying documents. But the attempt of natural language processing (NLP) has sought to model human language processing with some success. So this information extraction lies somewhere in between these two older endeavors, in terms of both difficulty and emphasis [36]. The common NLP tasks that are used in this paper are Tokenization, Part of Speech Tagging, Sentence Segmentation, Parsing and Named Entity Extraction.

2.2.1. Named Entity Recognition

Named Entity Recognition (NER) has been a well-studied problem for formal text and a subtask of Information Extraction. It has been introduced by Grishman and Sundheim to resolve the problem of identifying and categorizing the words into different classes. It is also known as Entity Identification, Entity Extraction or Entity Chunking. The term Named Entity is a word or phrase in a text that contains person's name, organization name, location, quantities etc. For example, we have a sentence "Ben works in Insight Corporation". Here "Ben" is the person's name and "Insight corporation" is the organization name in which Ben is working. So, NER is the process of classifying the entities based on some predefined categories like name, location, time, percentage etc. [3]. There is a vast research going on under this topic as NER is used in the preprocessing for most of advance NLP problems like information Extraction. So slowly there has been many tools that were developed to perform Named Entity Recognition like Stanford NER, Illinois NER and so on [5].

In this paper, we apply NER techniques to classify the signature from an email. There were lot of observation that can be done from the email corpus as how the signature would be. One important observation is that email messages generally contain some structured, easy-to-recognize signatures, which is nothing but the signature that will be directly appearing in a header and in a signature block. But that might not be the case in all emails as there would be complex terms that needs to be understood to classify whether that belongs to the signature block or not. So in this paper we present a novel approach where we are using three classification algorithms to deal with the extraction of signature from the email bodies.

2.2.2. Classifier

In machine learning, classification is the problem of identifying to which class or to which set of group the new data belongs to. There were different types of classifiers in machine learning based on their learning style. Some of them are Naïve Bayes classifier, Maximum Entropy classifier, Support Vector Machine, Decision Tree, Adaboost, Logistic Regression and so on. Some classifiers may belong to supervised learning style and some may belong to unsupervised learning models and each classifier has its own significance and it should be picked based on the problem or on the training set that we have. Here in our paper we are taking the classifiers based on supervised learning model.

2.3. Related Work

As email becoming the most advanced means of communication lot of research work has been done on this to extract the information from an email.

In 2004, Vitor R. Carvalho and William W. Cohen [6] addressed the problem of resolving signature references in the full email including the message body. They have used sequential and non-sequential algorithms to resolve this problem. In this they tried to detect if the email contains a signature block and then tries to extract the signature lines from the emails that contains the signature block. In this they applied features over the last k lines of an email to classify the signature and evaluated the results for different sequential and non-sequential algorithms. Identifying signature of an email user is important for name reference resolution and entity's identity modelling in emails.

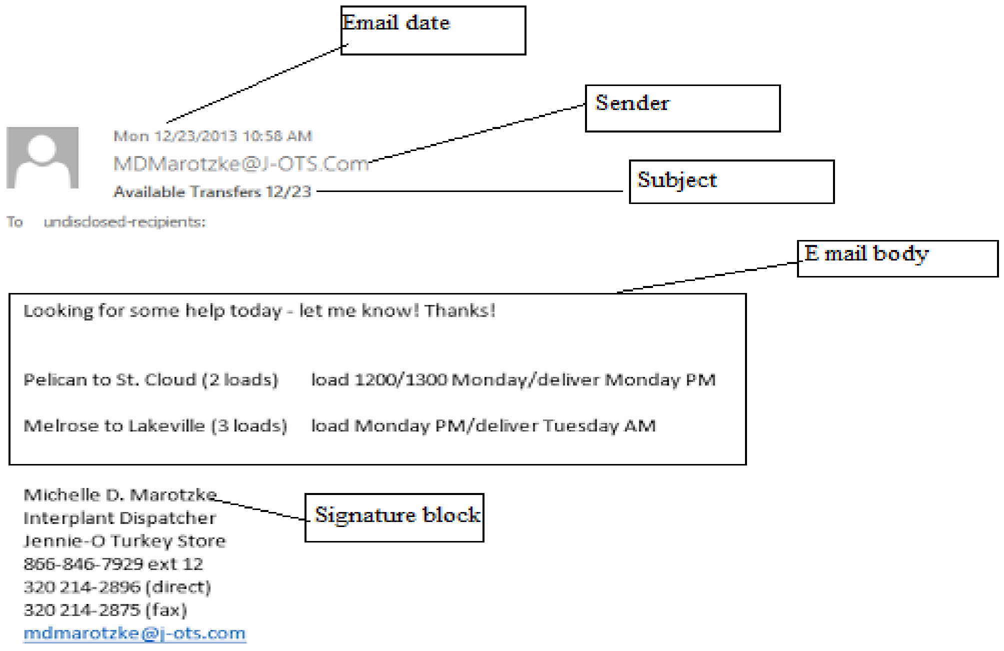
In 2004, there is another research that happened by Yuan Xiaoqin [1] where they tried to extract the signature and roles from the body of an email using unsupervised learning models. In 2006, K. Balog [7] studied the problem of correctly relating signature and email addresses that belong to the same entity by clustering. They extract (user name, address) pairs from the header of emails and cluster them by the similarity between the pairs. Neither of those two studies reported the difficult problem of extracting signature from email bodies.

In another study T. Elsayed et al. [8] in 2009, addressed the problem of resolving the personal name references from the large email collections including the message body. In this they tried to implement MapReduce Framework to find the names from the header, body and the signature.

In the paper that was published on 2011 by Meijuan Yin, Junyong Luo, Ding Cao, Xiaonan Liu and Yongxing Tan [9], they have considered about user name alias extraction from emails. It also directly relates to the extraction of signature from emails but only the user name from the signature, header and body are extracted. In this paper Named Entity recognition tools were used to solve this problem. As mentioned above, Named Entity recognition has the capability to identify the entities based on some predefined categories like person's name, location and so on. By using this approach, they extracted the user name from the email header, body and signature. In the recent study in 2015, there was a post that was published by Ryan [10] regarding the signature extraction importance. In that he mentioned about how the signature extraction project had uncovered 5,431 buried contacts and how those are useful for the company to establish contacts. Because of this project a signature database can be stored where there is a chance to access the hidden contacts of them which is really important for a client to establish relations with the other clients. Since, this is the kind of vast research that is still happening in this area, this paper is to propose a novel approach where signature extraction tool is implemented to extract the signature using supervised learning models.

CHAPTER 3. METHODOLOGY

3.1. Outline



Mon 12/23/2013 10:58 AM
MDMarotzke@J-OTS.Com
Available Transfers 12/23
To undisclosed-recipients:

Looking for some help today - let me know! Thanks!

Pelican to St. Cloud (2 loads) load 1200/1300 Monday/deliver Monday PM

Melrose to Lakeville (3 loads) load Monday PM/deliver Tuesday AM

Michelle D. Marotzke
Interplant Dispatcher
Jennie-O Turkey Store
866-846-7929 ext 12
320 214-2896 (direct)
320 214-2875 (fax)
mdmarotzke@j-ots.com

Fig 2: Email Format

We can clearly observe from the above figure about how the email message format looks like. An email message contains set of header fields which include from, To, Cc, Bcc, Subject and Date. And the rest of the message contains the body information. This body part may include the signature and also the quoted text if the original message is the reply message. So to analyze the signature and to extract it from the body we follow the following process. The first step we do is the preprocessing.

3.2. Preprocessing

The two sections header section and a body section constitute an email. All the general information is contained in the header section like email address of sender and recipient, subject of email as well as the route information whereas the actual message to be transmitted is contained in the body section. Preprocessing is a method by which this information is extracted prior to applying a filter process. The idea behind preprocessing is to convert messages in mail into a standardized format that is understandable to the machine learning algorithm. The sequence of steps that constitutes the process of preprocessing is given below: 1. Feature extraction: Extracting the features from e-mail like header or e-mail body. 2. Feature selection: Dimensionality reduction that is reduction of the vector of features. 3. Stop word removal: Removing the non-useful words. 4. Noisy removal: elimination of unwanted symbols or text from features 5. Features representation: depiction of features in an appropriate format for filtering [11].

3.2.1. Header and Body Extraction

To extract the header and body part we have an API called MAPI message parser class where it will identify from, To, Subject and the body part from an email message [12]. Once the body part is identified it is easy to detect the signature from the body based on some predefined patterns or tokens.

3.3. Signature Extraction

We use an email address to identify the corresponding signature as most of the signatures start with the sender’s formal name or informal names such as anonym, nickname, shortened form and so on. The target of our signature extraction system is extracting all the signature part present in the body from the given email corpus. The framework of our signature extraction system can be divided into three modules as shown in Fig.3: The first module is signature name Extraction in

Email Header. In this we take the email signature pairs of the sender and recipients from the email address fields such as “From”, “To”, “Cc” and “Bcc”. Next comes the second module Signature Block Identification. In this we locate whether the signature block is present in the body of an email. Once the signature block is identified, at last we try to extract the signature from the body and relate them to the email address that we got from the corresponding header of the email and generate new signature from the body [9].

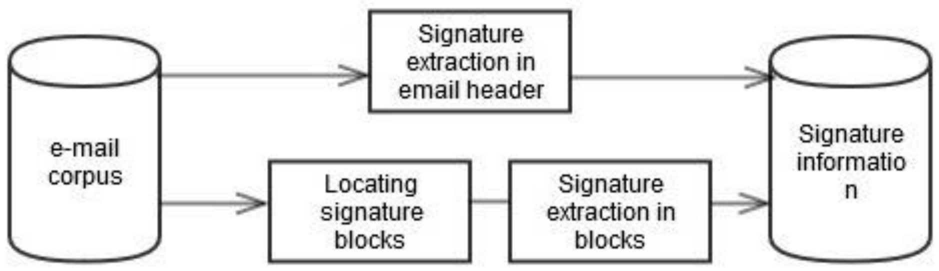

e-mail corpus
Signature extraction in email header
Locating signature blocks
Signature extraction in blocks
Signature informatio n

Fig 3: Framework of Email Signature Extraction System

3.3.1. Design of Signature Extraction System

Generally, in an email message, signature that can be directly associated with the email address of the email sender or recipient can be found in two locations. One is in email header fields such as “From”, “To”, “Cc” and “Bcc” in email headers where the alias of a user appears with the corresponding email address and the other one is salutation and signature blocks in email bodies, where the signature of a user appears.

In the Header part of the email, the user’s email address included by a pair of brackets, e.g. <email address>. So in the module of Signature Extraction in Email Header, we first extract the name from “From”, “To”, “Cc” and “Bcc” address fields in Header part by matching the punctuation pairs ‘ “ ’ and ‘ ” ’, and email addresses by matching the punctuation pairs ‘ < ’ and ‘ > ’, then associate each name with their respective email address and store it in a hash map which can be used later for the signature extraction from the email body.

In the Body part, the signature that is present in signature block can be directly related to the corresponding email address extracted by Module 1. A signature extracted from the signature block of an email body should be associated with the email address elicited from the “From” header of the same email, while a signature extracted from the salutation block of an email body can only be associated with the corresponding email address in the “To” header of the same email. However, the signature will not be in a fixed style in all the emails, it depends on the person’s interest on how he wants to keep the signature. So to extract the signature from the Body, we can use part-of-speech tagging tools to label block texts and identify candidate signature. And also named entity recognizing tools to get the aliases [9].

In this paper, the input for the signature extraction system will be the email corpus that we will be loading from the two types of datasets. And the output will be the signature part that is stored in a text file for each and every email. There are three ways that we can extract the signature from an email. One simple way is to directly fetch email name from “From, To, Cc and Bcc” headers which is used to extract the signature that contains the name at the beginning of the signature. The second one is to locate the signature block from the email body using some predefined features like name, location, organization, phone, fax and so on. The third one is using NER or part-of-speech tagging tools to label names in the signature blocks and obtaining candidate signature. In this way, we can finally extract all (email, signature) pairs and find all signature of each user appearing in the email corpus [9].

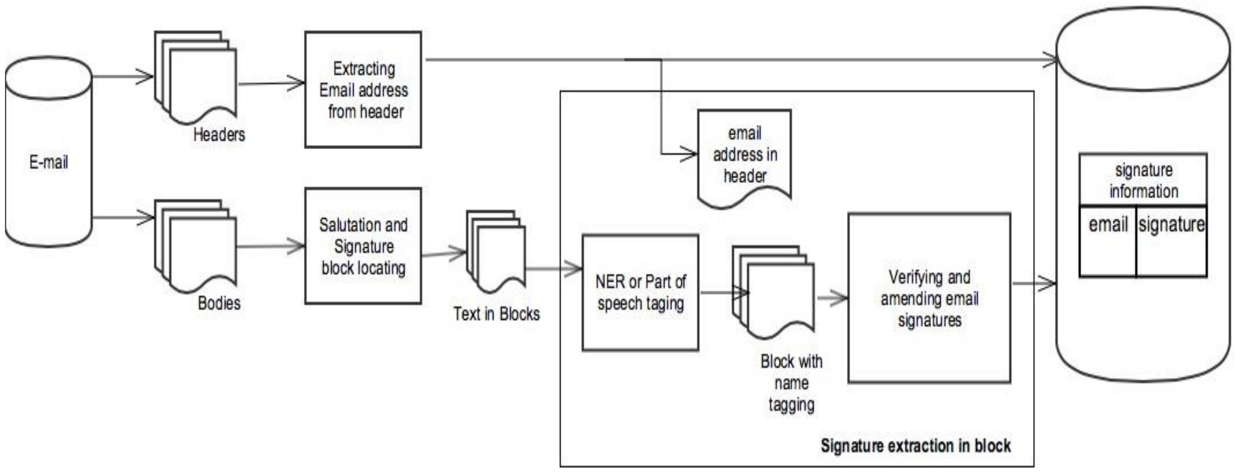

E-mail
Headers
Bodies
Extracting Email address from header
Salutation and Signature block locating
Text in Blocks
email address in header
NER or Part of speech taging
Block with name tagging
Verifying and amending email signatures
Signature extraction in block
signature information
email
signature

Fig 4: Flow of Signature Extraction System

3.3.2. Extracting Signature

In this part, we describe the methods to accurately extract signature from the salutation and signature block. We first briefly introduce the mechanisms which can be used to identify the feature set which in turn useful to extract the signature. Next we discuss about the algorithms that we are using to extract the signature, which is amply presented and evaluated in our former work.

3.3.2.1. Name Boundary

In the signature extraction system, after having located the signature blocks from email bodies, we use part-of-speech tagging tools to label block texts and identify names. There are some relatively mature part-of-speech tagging tools in different languages. As the English emails are considered in this paper, there is a well-known named entity recognition tool in English nature language process field called Named Entity Recognizer. The label of names tagged by NER is a pair of labels “<PERSON>” and “</PERSON>”, and between the pair of labels is a person name.

For example, a result tagged by NER is “<PERSON>James Johnson</PERSON>”, and the string “James Johnson” between the label pair “<PERSON>” and “</PERSON>” is an English person name.

By using above named entity recognizing tools, we can identify most of the format signature in salutation and signature blocks. However, all the names may not be formal and there may be some emails which have informal names such as nicknames, short names and so on. In such cases named entity recognizing tools may not identify the name. For example, the infrequent name “Li Shuo” is labeled as “Li/nr1 Shuo/ag”, based on which we can only extract the family name “Li”, and for the nickname “Little Dou” tagged as “Little/a Dou/n”, we can’t extract any part of the nickname [9]. Based on the above factors, named entity recognizer alone can’t extract the exact and accurate signature and we need to consider some more mechanisms to do it.

The other way to extract the signature is based on the keywords present in the body before signature part appears such as “Thanks”, “Regards”, “Yours truly”, “Sincerely”, “Best Regards” and so on, and with a space or an end symbol of the text line after them. So in this paper, we also consider this feature of words to improve the accuracy of extracting signatures from the signature block. We build a name boundary word template for general use based on the above feature of words around names in email salutation and signature blocks, and extract the signature block from the body of an email with the use of the above keywords.

3.3.2.2. Tokenization

It is the process of breaking up the given text into units called tokens. The tokens may be words phrases or number or symbols or any other meaningful elements [14]. Tokenization does this task by locating word boundaries. Ending point of a word and beginning of the next word is called word boundaries. Usually Tokenization is considered as a preprocessing step. And there are some steps that we go through in this Tokenization

- A simple and most used method in majority of applications is segmenting text into words. In this it splits the text into words by removing the spaces and the punctuation marks.
- Next step that we undergo is handling abbreviations. For this abbreviations should be maintained in a dictionary such that during the tokenization a word with a trailing period can be looked up and if it is found then it will take it as a single token.
- Next step is handling of any numerical or special expressions. We can mention some examples like email address, telephone number, URLs, dates, time and so on.
- There is one more step where we try to handle the hyphenated words. Hyphenation can be used for various purposes like to split the vowels in words like Co-Education and to join the nouns as names and so on. Generally hyphenated words are treated as a single unit and prefer to divide into a single token [32].

Once this tokenization is done, these list of tokens will be the input for further processing such as parsing and Once the extraction process is done by using these methods body and signature data is used by a technique called Hash Map.

3.4. Hash Technique Mechanism

Hash Map is a data structure that can map keys to values to store the object and retrieve with the help of key value pairs [18]. In hash technique, Objects are stored by calling put (key, value) method of Hash Map and retrieved by calling get (key) method [19]. When we call put method, hash code () method of the key object is called so that hash function of the map can find a bucket location to store value object, which is actually an index of the internal array, known as the table. Hash Map internally stores mapping in the form of Map**.** When we want to retrieve the object, we call the get () method and pass the key object to retrieve that specific value. When a key is passed it generates same hash code as hash map keys are immutable and based on that we end up at same bucket location from where we want to retrieve the value. If there is only one object at that specific location, then it is returned and that is the value object which we have stored earlier. Since the internal array of Hash Map is of fixed size, and if we keep storing objects, at some point of time hash function will return same bucket location for two different keys and we call this as collision in Hash Map. In this case, linked list will be used and a new entry will be stored at the next node. In this case of linked list when we try to retrieve an object, we need an extra check to search correct value, this is done by equals () method. Since each node contains an entry, Hash Map key compares the entry key object with the passed key using equals () method and when it returns true, Map returns its corresponding value [31].

3.5. Training Classifiers

To classify the signatures, we have to create a Trial Object passing the Classifier name and the Instance List (input data).

```
for (int i = 0; i < trainers. Length; i++) {// perform this for all trainers
        result = new Result Bean ();
```

```
                Trial t = new Trial(classifiers[i], testing); // create the trail

                result. set Name(classifiers[i]. getClass (). getName ());

                -------

                results. Add(result);

        }
```

When we create the Trial object, with a specified classifier this will classify the data.

```
        public Trial (Classifier c, InstanceList ilist)

        {

                super (ilist. size ());

                this. Classifier = c;

                for (Instance instance: ilist)

                this.add (classify (instance));

        }
```

In this method, we classify the given input data according to the respective classifier name. We have followed the Factory design pattern to construct the specific algorithm classifier objects. The Main class is Classifier, which is extended by Naive Bayes classifier, Maximum Entropy classifier, and Support Vector Machine classifier which overrides the Base class behavior of classification.

There are 3 types of learning models namely Supervised, Unsupervised and Semi-supervised learning. In Supervised learning model, input data is known and we need to make the predictions based on that. In Unsupervised model, input data is not labelled and it does not contain a known result as in supervised model. Semi-supervised learning model is nothing but the combination of supervised and unsupervised model i.e. it contains mixture of labelled and unlabeled data. Here in our project for the signature extraction we have considered three methods namely Naïve Bayes, Maximum Entropy and Support Vector Machine where all the three are under supervised learning model as we already have the trained data nothing but the input data and we are predicting the results based on the input.

3.5.1. Naïve Bayes Classifier

The naïve Bayes classifier is a probabilistic classifier which is based on Bayesian theorem. It operates on a strong independence assumption [20]. This means that the probability of one attribute does not affect the probability of the other. Given a series of n attributes, the naïve Bayes classifier makes 2n! independent assumptions. Nevertheless, the results of the naïve Bayes classifier are often correct. In [21] the author observes some conditions under which the naïve Bayes classifier performs well and why. They state that the error is affected by three factors: training data noise, bias, and variance. The only way to remove training data noise is only by choosing good training data and the training data must be divided into several groups by the machine learning algorithm. Next one is the bias which occurs due to large groupings in the training data. Lastly, variance is the error due to those groupings being too small [22].

Using Bayes theorem, we write the equation as [wiki]

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Here A and B are the events, P(A) and P(B) are the probabilities of both A and B. P (A | B) and P (B | A) are the conditional probabilities of an event given that the other event is true.

Naïve Byes is one of the most basic and simple text classification technique. Despite of its simple design, it is one of the most essential algorithm and used in many real-world problems. It is involved in several applications like email spam detection, personal email sorting and so on [23].

Naive Bayes classifier is most efficient in terms of CPU and memory and also it requires less training data. Moreover, the training time that Naive Bayes algorithm takes is smaller when compared to the other classifiers. So usually when the training time is a key factor, Naïve Bayes is the first algorithm that comes to mind as it can be trained quickly. In 2008, Manning et al proved that, even though the probabilities of Naïve Bayes classifier are of low quality, the decision made by this classification technique are quite good [23]. As Naïve Bayes over estimates the probability, we use this classifier to make the decision but not to accurately predict the exact probabilities and hence by this we can say the model is accurate as it can make the correct decisions.

Generally, in a text classification problem, we use the tokens to classify that particular text in the document based on the appropriate class. In our case, we need to classify the signatures from the email corpus based on some tokens. By using the maximum a posteriori (MAP) decision rule, we come up with the below equation as in [23], where it will be useful in our project to detect the signature patterns:

$$c_{map} = \arg\max_{c \in C} \left(P(c \mid d) \right) = \arg\max_{c \in C} \left(P(c) \prod_{1 \le k \le n_d} P(t_k \mid c) \right)$$

Here t_k are the tokens of the email corpus, C is the set of classes that is used in the classification, P(C | D) is the conditional probability of class c given document d, P(C) is the prior probability of class c.

The above statement means that in order to find the class, we should have an estimate of the probability of each word of that class called likelihood, multiplied by the probability of the prior class. After calculating the above equation for all the classes, we can select the highest probability class.

3.5.2. Maximum Entropy Classifier

The Max Entropy classifier is a probabilistic classifier which is based on the principle of Max Entropy. Principle of Maximum Entropy states that when estimating the probability distribution, we should select the distribution that has the largest entropy [24]. As discussed above about the naïve Bayes that it assumes the features are independent. But in this case Max Entropy does not assume that the features are independent. The Max Entropy classifier is used in the natural language processing and it is widely used in solving the text classification problems, information retrieval problems. Some of the problems include language detection, topic classification, sentiment analysis in which this classifier is widely used [29].

Maximum Entropy classifier is used whenever we don't know anything about the prior distributions, in the cases where we can't make any assumptions and when we can't assume the conditional independence of the features. But we consider the Maximum Entropy classifier in our project as we are dealing with the signature classification where all the features are the words which most likely are not independent. When we consider training Maximum Entropy classifier, it takes more time to train as it follows the optimization problem in which it needs to estimate the parameters which is not the case in Naïve Bayes classifier. Here the optimization problem in the sense it uses search mechanism to find the set of parameters which requires more time but this method provides robust results and also it is quite good in terms of CPU and memory consumption [29].

<u>How it works</u>

Maximum Entropy Model uses search based optimization techniques in iterative way to find the weights for the features that maximizes the likelihood of the training data. Initially it takes the parameters to random values through search criteria then it refines those parameters in such a way so that it will be near to optimal solution. But this entire process takes long time to learn if the training data is large so that is the reason we say maximum entropy classifier takes more time to train the data. We can define Maximum Entropy classifier through this equation [33]

$$P(c|d,\lambda) \overset{def}{=} \frac{\exp \sum_i \lambda_i f_i(c,d)}{\sum_{c' \in C} \exp \sum_i \lambda_i f_i(c',d)}$$

Where C is the class in the document D and λ is the weight that we give for each feature to know the maximum likelihood of the training data. Firstly, in this model, it tries to take the word list from the class C, and for each word W, it assigns a weight to joint feature f (w, c) nothing but how many times the word appeared in that particular class in a document. Once we calculate, we simply apply the above equation to calculate maximum likelihood for that particular class C in a given document D.

3.5.3. Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. These are based on the concept of decision planes. A decision plane is one that separates between two different set of objects having different classes. Basically SVM performs the classification tasks by constructing the hyperplanes in a multidimensional space based on different class labels. These hyperplanes are constructed in such a way that they make a good separation between two different set of objects which we call it as a functional margin. The larger is the margin the lower is the error. [25]

In SVM**,** we plot each data item as a point in n-dimensional space where n will be the number of features with respect to the coordinates. Then we perform a classification by constructing a hyper plane which separates two different classes. Suppose we have two objects one is green and the other is red. The hyperplane is the separating line which defines a boundary and to the right side of it contains the green objects and to the left all objects will be red. And we want to classify a new object and if it falls to the right then it will be labelled as green and to the left will become red. This will be decided based on the support vectors which are the co-ordinates of individual observation [28].

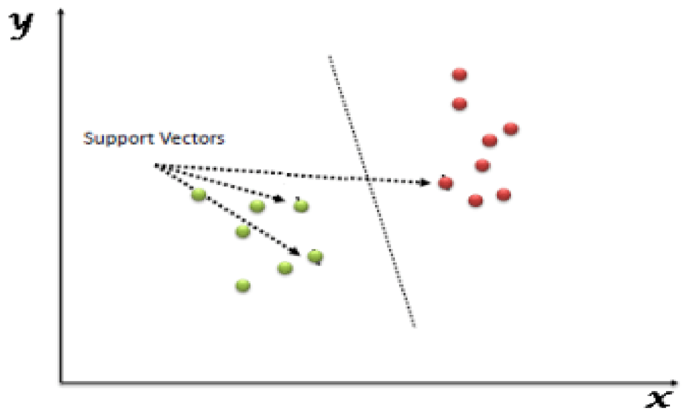


Fig 5: SVM Boundary Segregation

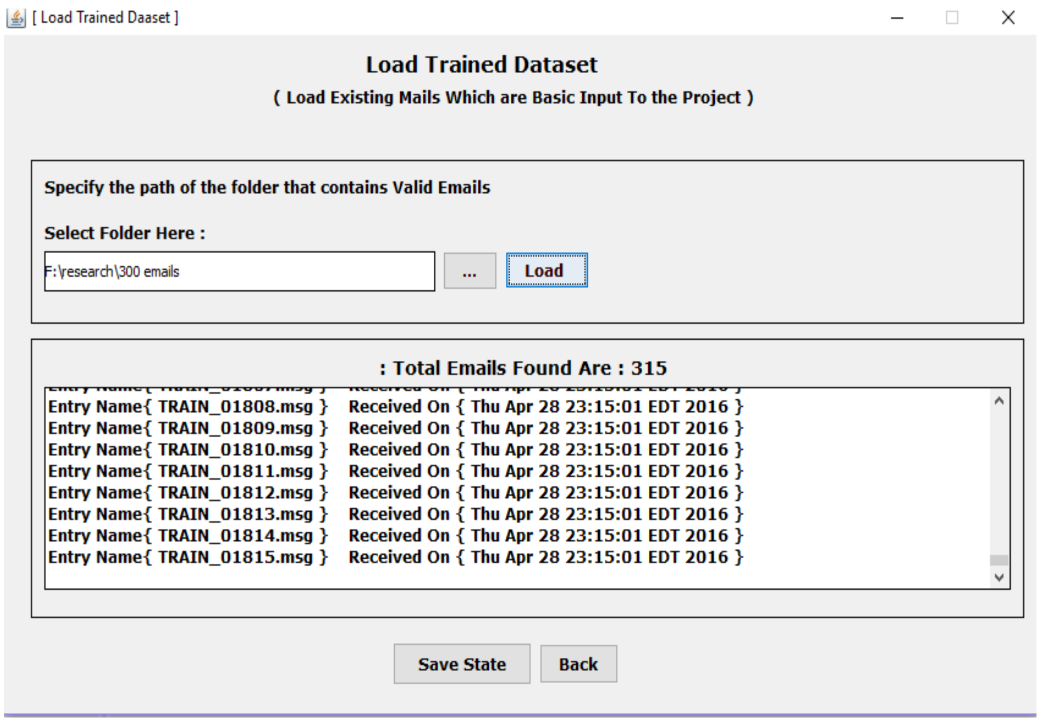
From the above SVM Boundary segregation, the middle line that separates the red and green objects is the hyperplane and points which are near to the line are the support vectors that shows the margin nothing but the distance to that hyperplane from the both sides of the support vector objects. SVM follows one technique called the kernel trick in which it takes low dimensional input space and transform it to a higher dimensional space. This is the technique that is used in non-linear problem. As our project is based on the text classification, to start training our SVM model firstly we take the data from the email corpus. Then we divide the email into set off words and generate the vocabulary based on it. Finally, we create a document term matrix. A document term matrix is nothing but a mathematical matrix which describes the frequency of words that occur in the document. So, this matrix prepares the words and how many times it appears.

3.6. Implementation

Signature Extraction Tool is implemented in Java using three supervised machine learning algorithms namely Naïve Bayes Classifier, Maximum Entropy Classifier and Support Vector Machine Classifier. Signature Extraction tool has three phase of extracting the signature form emails. They include Loading the email corpus, Extracting the signature from that Enron corpus and evaluating the results.

3.6.1. Dataset Loader

This is the first phase in our tool where we try to load the emails from the two datasets that we are considering in this paper and we save these emails in the session, so that we can use this in our extraction phase to extract the signature from the email bodies.


[Load Trained Daaset]
Load Trained Dataset
(Load Existing Mails Which are Basic Input To the Project)
Specify the path of the folder that contains Valid Emails
Select Folder Here :
F:\research\300 emails
...
Load
: Total Emails Found Are : 315
Save State
Back

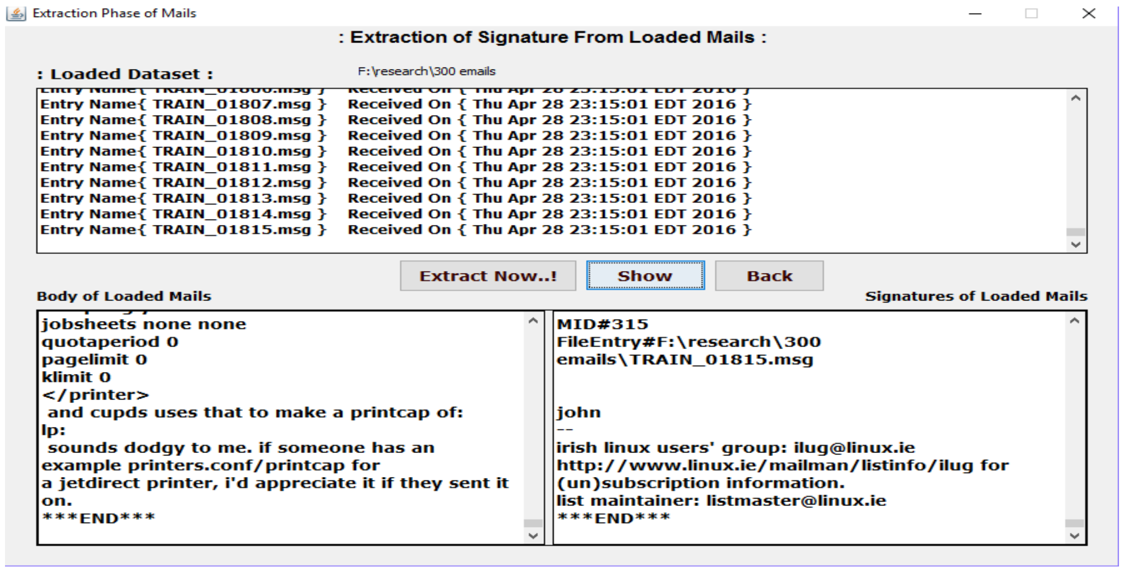
```
Entry Name{ TRAIN_01808.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01809.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01810.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01811.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01812.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01813.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01814.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01815.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
```

Fig 6: Dataset Loader

3.6.2. Extraction Phase

In this phase we try to extract the header and signature from the email bodies. To extract the header, we use a package MAPI message parser class where it contains some default methods to extract the cc, body, subject, from and to from the email. To extract the signature from the body we follow some feature set based on some NLP mechanisms and based on those features we extract the signature lines from the emails. The features that we have included in our project for signature extraction are

- If the email contains Email pattern
- If it contains Name
- If it contains Phone Number Pattern
- If it contains punctuations
- If it contains special symbols, Case and Tabs
- Regular expressions can be used in the cases where it contains complex words.
- We use position features to determine what is the position the name is starting and based on the we can identify the signature as mostly signature starts with the name.



: Extraction of Signature From Loaded Mails :

: Loaded Dataset : F:\research\300 emails

```
Entry Name{ TRAIN_01807.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01808.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01809.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01810.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01811.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01812.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01813.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01814.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
Entry Name{ TRAIN_01815.msg }   Received On { Thu Apr 28 23:15:01 EDT 2016 }
```

Body of Loaded Mails

```
jobsheets none none
quotaperiod 0
pagelimit 0
klimit 0
</printer>
 and cupds uses that to make a printcap of:
lp:
 sounds dodgy to me. if someone has an
example printers.conf/printcap for
a jetdirect printer, i'd appreciate it if they sent it
on.
***END***
```

Signatures of Loaded Mails

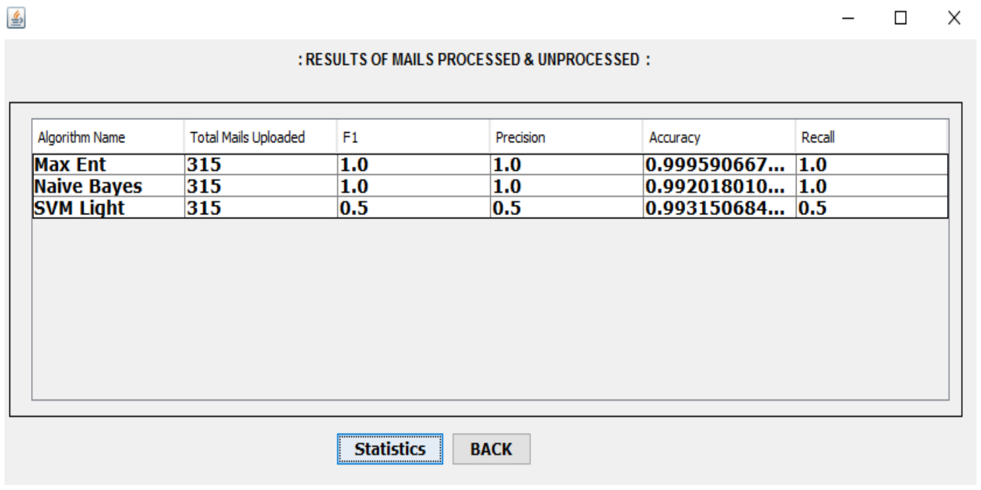
```
MID#315
FileEntry#F:\research\300
emails\TRAIN_01815.msg


john
--
irish linux users' group: ilug@linux.ie
http://www.linux.ie/mailman/listinfo/ilug for
(un)subscription information.
list maintainer: listmaster@linux.ie
***END***
```

Fig 7: Signature Extraction

3.6.3. Result Phase

This is the last phase where we try to apply the three algorithms that we are considering in this paper. Later we try to evaluate these features based on these algorithms and try to calculate the performance metrics for all the three algorithms. In this phase we are calculating Precision, Recall, F1 Measure and Accuracy.



: RESULTS OF MAILS PROCESSED & UNPROCESSED :

Algorithm Name	Total Mails Uploaded	F1	Precision	Accuracy	Recall
Max Ent	315	1.0	1.0	0.999590667...	1.0
Naive Bayes	315	1.0	1.0	0.992018010...	1.0
SVM Light	315	0.5	0.5	0.993150684...	0.5

Fig 8: Result Set

CHAPTER 4. EVALUATION

4.1. Experimental Design

4.1.1. Dataset

We analyze and validate the availability of our method to extract the signature from the emails. For this we have considered two datasets. One is the dataset from the valley Express Company which consists of 1000 emails. These emails contain the communication that are sent among employees and the customers of the Enron. And the other dataset is the public dataset, CSDMC2010 corpus which consists of 2000 emails [26]. These are the two datasets that have large-scale mail archives with different formats of signature. Hence it would be a good evaluation technique to find the feasibility of our algorithms and hence shows that experimental results of our signature blocks locating algorithms on these emails have shown a relatively high performance.

4.1.2. Evaluation Metrics

In our experiment, we use three standard Evaluation Metrics, Precision, Recall, F1 to evaluate the performance.

4.1.2.1. Precision

Precision represents the fraction of retrieved items that are relevant which is nothing but the number of correct results delivered divided by the number of all items retrieved [27]

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

4.1.2.2. Recall

Recall represents the fraction of relevant items that has been retrieved which is nothing but the number of correct results achieved divided by the number of correct results that were supposed to be returned [27].

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

4.1.2.3. F1 Measure

A measure that combines precision and recall is the harmonic mean of precision and recall. This is known as F1 measure because recall and precision are evenly weighted [27].

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

4.2. Experimental Results

For Valley Express dataset, we have randomly selected 1000 emails and performed extraction using three algorithms Naïve Bayes, Maximum Entropy and Support Vector Machine. Based on the evaluation metrics we have calculated the performance for three of the algorithms. In the similar manner, we have considered 1500 emails randomly from the public dataset and evaluated the performance based on precision, Recall, F1 Measure.

4.2.1. Comparison

The result is compared among the three classifiers for determining its efficiency. The robust classifiers namely Naïve Bayes, Maximum Entropy and SVM are tested with the same dataset of emails and their results obtained are summarized in the form of table for comparison with another classifier. The following table shows the comparison of classification efficiency of the various classifiers and the number of classified instances by various classifiers out of the total number of instances. These values are obtained by applying all the classifiers one by one on the same dataset of emails.

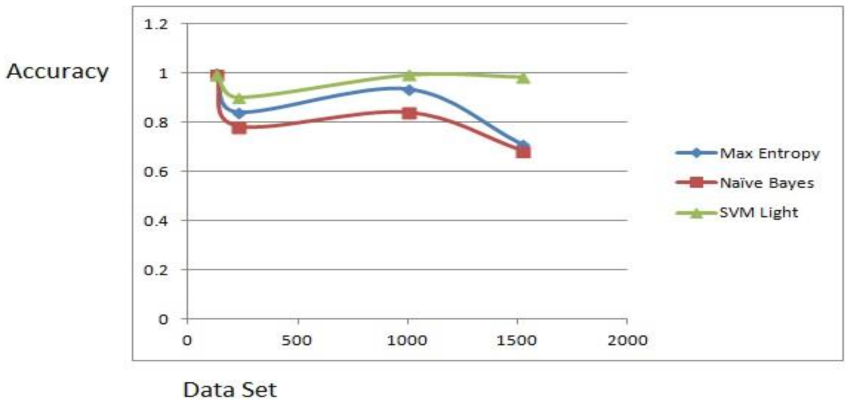


Fig 9: Number of Datasets Vs Accuracy

From the above graphs we can evaluate the performance for the three algorithms. Fig 1 shows the accuracy of the three algorithms in which a generic dataset is applied for the signature extractions. And from this fig we can know that SVM has the highest accuracy when compared to the other three algorithms. The accuracy can be stated as the number of correct predictions from all the predictions made. On a rough estimate if we take the accuracy of the three classifiers the best accuracy achieved is 99%,98%,94% for SVM, Max Entropy and Naïve Bayes respectively.

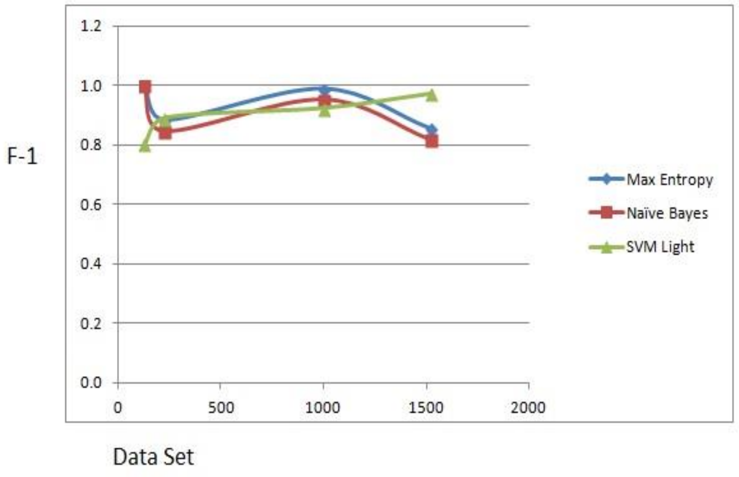


Fig 10: Number of Datasets Vs F1

Fig 2 shows the F1 measure The best F1 measures we achieve for Maximum Entropy, Naive Bayes, and SVM are 88.83%, 88.71%, and 80.72% respectively. F1 measure simply means the harmonic mean of precision and recall. The better the precision and recall values, the better F1 measure will be. Here from the graph The F1 value evaluated on our datasets were quite good from which we can say that all these three algorithms have enough scalability to perform signature extraction on large-scale mail archives.

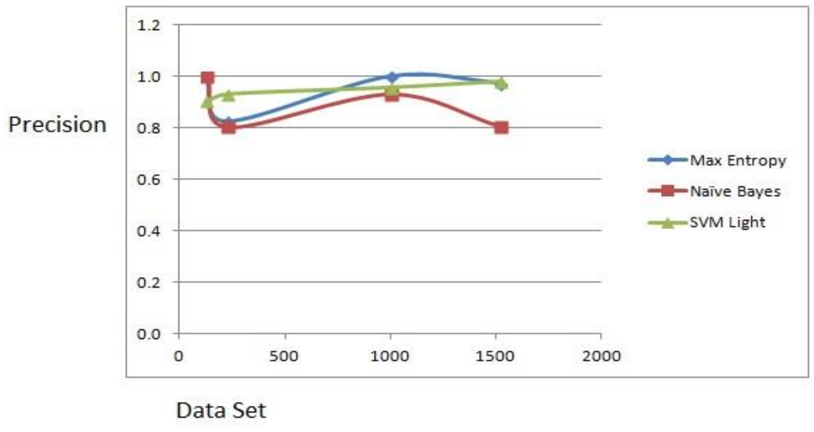


Fig 11: Number of Datasets Vs Precision

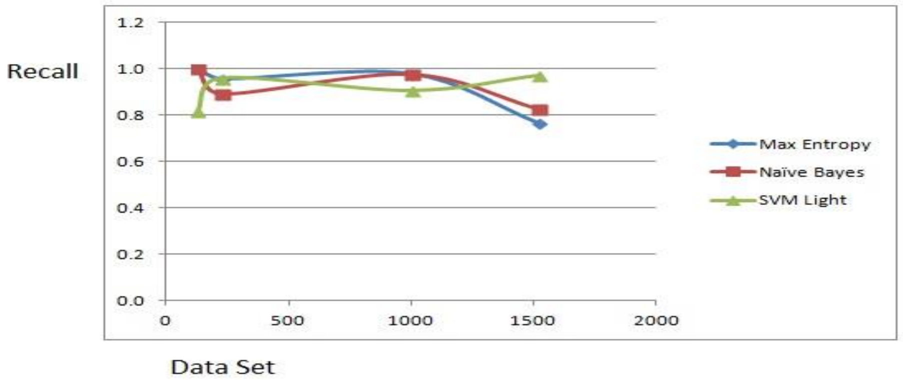


Fig 12: Number of Datasets Vs Recall

Fig 3, 4 shows the precision and recall values respectively. Precision is nothing but the number of correct items retrieved. Precision and recall are exactly the opposite sides of the performance. Both combined together forms an integrated F1 measure which is very important evaluation metric based on which we can decide how much the algorithm is effective. The highest precision and recall we can observe for the Max Entropy classifier. The extremely low recall can be explained by the fact that in the informal texts, it is very likely that true names do not appear in a complete sentence. The best recall measures for Maximum entropy, Naive Bayes and SVM are 81.67%, 74.18%, and 63.91%, respectively. From the result of our experiment, we find in the most cases, recall is lower than precision, and thus lower the F1 metric. To improve the relatively low recall, we can try to include more datasets features that contain the information throughout a single message or across the multiple messages instead signatures that is constrained in a sentence. For example, a signature may occur multiple times in a corpus, especially in email corpus which is associated with a group that works closely together. Therefore, information of a signature repetition could be beneficial to recognize the one in an ambiguous context.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1. Conclusion

In this paper we addressed the problem of automatically extracting signature of email users from the full email message. For this, we have used an approach where we have used the algorithms namely Naïve Bayes, Maximum Entropy and SVM. Using this, we were able to extract the whole signature part from the email message. And we have compared the results among these three algorithms. All the three algorithms have its own effective features and purely depends on how that algorithm is used on our dataset. Experimental results on the email dataset clearly shows that our approach in which we have used the three algorithms are effective in accuracy and performance which is very important in real time projects.

5.2. Future Work

As the email signatures will be having different formats, our future work include working on different formats of signatures especially the image format. Digital image format is totally different and requires unique approach which we will be considering the number format to deal with this problem. Hence we can consider this as our future task. And also we can investigate novel methods that can be applied to the full email for identifying the pseudo signature.

REFERENCES

[1] Y. Xiaoqin "Unsupervised Extraction of Signatures and Roles from Large Scale Mail Archives" published in International Journal of Security and its Applications, Vol. 9, No. 4, pp. 229-238, April 2015

[2] Prakash M Nadkarni, Lucila Ohno – Machado and Wendy W Chapman "Natural Language Processing: An Introduction" v. 18(5); Sep- Oct 2011, Retrieved from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328/>

[3] "Named Entity Recognition." Retrieved from <https://en.wikipedia.org/wiki/Named-entity_recognition.>

[4] Yu-shan Chang and Yun-Hsuan Sung, "Applying Name Entity Recognition to Informal Text" < https://pdfs.semanticscholar.org/50b5/d0dd6ffff792e43321f09fced07f0cf3d4d0.pdf>

[5] Qianzhou Du, Xuan Zhang, "Named Entity Recognition" <https://vtechworks.lib.vt.edu/bitstream/handle/10919/52254/ReportNER.pdf?sequence=15&isAllowed=y>

[6] V. Carvalho and W. Cohen, "Learning to extract signature and reply lines from email", Proceedings of the Conference on Email and Anti-Spam, (2004).

[7] Krisztian Balog and Maarten de Rijke, "Finding Experts and their Details in E-mail Corpora", ISLA, University of Amsterdam, <https://krisztianbalog.com/files/www2006-expertmail.pdf>

[8] TM Elsayed, Doctor of Philosophy, 2009, "Identity Resolution in Email Collections", <https://drum.lib.umd.edu/bitstream/handle/1903/9618/Elsayed_umd_0117E_10665.pdf?sequence=1&isAllowed=y>

[9] Meijuan Yin, Junyong uo, Ding Cao, Xiaonan Liu and Yongxing Tan, "User Name Alias Extraction in Emails", Published in April 2011 in MECS

[10] Email Signature Extraction, Retrieved from <https://www.linkedin.com/pulse/email-signature-extraction-project-uncovers-5431-buried-ryan-murray>

[12] Apache POI, Retrieved from <https://poi.apache.org/apidocs/org/apache/poi/hsmf/MAPIMessage.html>

[13] Mansi Goyal and Ankita Sharma, "An Efficient Malicious Email Detection Using Multi Naïve Bayes Classifier" Volume 5, Issue 5, May 2015 <https://www.ijarcsse.com/docs/papers/Volume_5/5_May2015/V5I5-0343.pdf >

[14] Tokenization, Retrieved from <https://en.wikipedia.org/wiki/Tokenization_(lexical_analysis)>

[15] Art of Tokenization, Retrieved from <https://www.ibm.com/developerworks/community/blogs/nlp/entry/tokenization?lang=en>

[16] “Extracting Personal Names from Email: Applying Named Entity Recognition to Informal Text”, Retrieved from < https://www.cs.cmu.edu/~einat/email.pdf>

[17] David Nadeau and Satoshi Sekine, “A survey of Named Entity Recognition and Classification” National Research Council Canada/New yok University, Retrieved from <https://nlp.cs.nyu.edu/sekine/papers/li07.pdf>

[18] Hash Table, Retrieved from <https://en.wikipedia.org/wiki/Hash_table>

[19] Hash Map, Retrieved from <https://javarevisited.blogspot.com/2011/02/how-hashmap-works-in-java.html>

[20] Naïve Bayes Classifier, Retrieved from <https://en.wikipedia.org/wiki/Naive_Bayes_classifier>

[21] P. Domingos, and M.J. Pizzani, "On the optimality of the simple Bayesian classifier under zero-one loss”, m/clearning, Vol.29, no2-3, pp 103-130, 1997 <https://www.academia.edu/6689205/NETWORK_INTRUSION_DETECTION_USING_NA%C3%8FVE_BAYES>

[22] Mrutyunjaya Panda and Manas Ranjan Patra, “Network Intrusion Detection Using Naïve Bayes” Vol. 7 No. 12, December 2007 <https://pdfs.semanticscholar.org/1a5c/191da4aa733c80311ef4057c16dc899819cd.pdf>

[23] Naïve Bayes Text Classifier, Retrieved from < https://blog.datumbox.com/machine-learning-tutorial-the-naive-bayes-text-classifier/>

[24] Maximum Entropy, Retrieved from <https://en.wikipedia.org/wiki/Maximum_entropy_probability_distribution>

[25] Support Vector Machine, Retrieved from <https://en.wikipedia.org/wiki/Support_vector_machine>

[26] Email Datasets, Retrieved from <https://csmining.org/index.php/spam-email-datasets-.html>

[27] Precision and Recall, Retrieved from <https://en.wikipedia.org/wiki/Precision_and_recall>

[28] Support Vector Machine, Retrieved from <https://www.analyticsvidhya.com/blog/2015/10/understaing-support-vector-machine-example-code/>

[29] Maximum entropy Text Classifier, Retrieved from <https://blog.datumbox.com/machine-learning-tutorial-the-max-entropy-text-classifier/>

[30] History of Internet, Retrieved from <https://en.wikipedia.org/wiki/History_of_the_Internet>

[31] Hash Map, Retrieved from <https://www.quora.com/How-is-Hashmap-in-Java-implemented-internally-What-are-the-pros-and-cons-to-use-it-What-are-the-complexities-it-provides-for-insert-delete-and-lookup>

[32] Tokenization, Retrieved from <https://nlp.stanford.edu/IRbook/html/htmledition/tokenization-1.html>

[33] Maximum Entropy, Retrieved from <https://sentiment.christopherpotts.net/classifiers.html>

[34] F. Angiulli, T. Catarci, P. Ciaccia, G. lanni, S. Kimani, S. Lodi, M. Patella, G. Santucci and C. Sartori, "An Integrated Data Mining and Data Presentation Tool", Published in 2002 in WITPress

[35] Data Analysis and Data Mining, Retrieved from <https://www.dbta.com/Editorial/Trends-and-Applications/What-is-Data-Analysis-and-Data-Mining-73503.aspx>

[36] Natural Language Processing, Retrieved from <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.41.5985&rep=rep1&type=pdf>