Massive Enhanced Extracted Email Features Tailored for Cosine Distance

Farshad Barahimi

Dalhousie University, Canada

(Farshad Barahimi is a PhD student at Dalhousie University)

Abstract

In this paper, the process of converting the Enron email dataset to thousands of features per email for a selected set of 2400 labelled emails is explained and evaluated. The final features are tailored for Cosine distance so that the Cosine distance inversely reflect the number of top indicative words of each email that are common between the two emails in an explainable normalized fashion. The labelling is based on the leaf folder name in the Enron email dataset folders tree and the 2400 emails selected consist 300 emails for each of the 8 labels. The evaluation is based on the accuracy of a $k$ nearest neighbours majority voting classification using Cosine distance. In addition to KNN majority voting classification accuracy and confusion matrix, some statistics for the process is reported. The KNN majority voting classification accuracy using Cosine distance is 76.75% which shows at least some level of success given the 8 labels involved. The result of conversion is 48557 features per selected email out of which exactly 40 features per email are non-zero. The result of conversion is a data set named MeeefTCD (Massive Enhanced Extracted Email Features Tailored for Cosine Distance) available at https://web.cs.dal.ca/~barahimi/data-sets/meeeftcd/ and on a github repository mentioned in this paper.

I. INTRODUCTION

The Enron email dataset, is a publicly available data set of Enron emails which was originally prior to modifications made public and posted online by the United States Federal Energy Regulatory Commission during an investigation and later a modified version of it was posted online by Cohen. The modified version labelled May 7, 2015 and posted online by Cohen is used in this paper to produce email features for a selected set of 2400 emails. The emails in Enron email dataset are organized in folders which in some cases are reasonable choice for labelling emails.

While different methods exist to measure dissimilarity of text documents in order to better organize the documents, the Cosine
distance of Term Frequency–Inverse Document Frequency is a common method but criticized \cite{2,3}. In short, in a common approach a text document such as an email is seen as a vector of constant number of numbers each relating to the importance of a word of a fixed dictionary for that document. Then to measure distance of two text documents specified by two vectors \( V_a = (v_{a1}, v_{a2}, \ldots, v_{am}) \) and \( V_b = (v_{b1}, v_{b2}, \ldots, v_{bm}) \), the Cosine distance of the two vectors denoted by \( \theta \) is used which is equal to:

\[
1 - \frac{\sum_{1\leq i \leq m} (v_{ai} \cdot v_{bi})}{\sqrt{\sum_{1\leq i \leq m} v_{ai}^2} \cdot \sqrt{\sum_{1\leq i \leq m} v_{bi}^2}}
\]  

(1)

An important observation which is the key piece of this paper is that if in the formula above (Equation 1) there is a restriction that the value of each \( v_{ai} \) or \( v_{bi} \) is restricted to be either 1 or 0 (binary), and if the number of ones in \( V_a \) is the same as the number of ones in \( V_b \), then Cosine distance essentially acts like a digital \textit{and} operator counter which counts the number of places where both vectors have ones normalized by a constant factor. This is important to have a meaningful explanation of the meaning of the Cosine distance. The two other important factors to consider in order to have a meaningful explanation of the meaning of Cosine distance is the meaning of each 1 and the number of 1’s in each vector. This paper strives for the meaning of a 1 to be being among top indicative words for that email where definition of top indicative words is as defined in this paper in Section \textbf{II} and subsequently since the number of 1’s won’t be fixed as desired, some secondary 1’s are added in extra vector dimensions (increasing \( m \)) to have a fixed the number of 1’s for each of all vectors. This changes each vector \( V_a \) to a vector \( \hat{V}_a = (v_{a1}, v_{a2}, \ldots, v_{am}, v_{a(m+1)}, \ldots v_{aq}) \) where the first \( m \) dimensions are called primary dimensions and the next \( q \) dimensions are called secondary dimensions. The desired number of 1’s per vector is based on some statistical analysis described in Section \textbf{III} however it is important that when a dummy 1 is added to a vector in a secondary dimension, the other vectors don’t have a one in that dimension, so it only affects normalization but not counting. While the first \( m \) dimensions help counting, the next \( q \) dimensions help with consistent constant normalization.

To measure the importance of a dictionary word for a text document, a common method is Term Frequency–Inverse Document Frequency \cite{2,3} which has some variants but in this paper the variant defined by the following formula is used:

\[
v_{ai} = \frac{F_{a_i}}{\sum_t F_{at}} \cdot \log\left( \frac{N}{\hat{F}_{i}} \right)
\]

where \( F_{a_i} \) is the number of times that the \( i^{th} \) word of dictionary appears in the email
Having this introduction, Section II goes through the details of generating features for emails that are tailored for Cosine distance. It is important to note that this introduction missed some of important details and steps that are described in Section III. In Section IV, not only accuracy of a KNN majority voting classification is reported but also the confusion matrix is discussed along some statistics. Finally Section V talks about how the data set is available and also where to find the code used to create the data set and produce the results of this paper.

II. FEATURES GENERATION PROCESS

A. Selecting emails

The first 300 emails from each of 8 folders labelled projects, logistics, resumes, universities, online_trading, meetings, management and ces are selected. Due to the nature of the Enron email dataset [1], it is possible that multiple folders are labelled the same, in which case only the first one in file traversal of the emails root folder which passes the criteria of having at least 300 emails is chosen. In this paper, the ordered set of selected emails are denoted by $E = (e_1, e_2, ..., e_{2400})$. For each of the labels a number between 0 and 7 is used in this paper and the final data set (MeeefTCD) as shown in Table I.

| label            | label number |
|------------------|--------------|
| projects         | 0            |
| logistics        | 1            |
| resumes          | 2            |
| universities     | 3            |
| online_trading   | 4            |
| meetings         | 5            |
| management       | 6            |
| ces              | 7            |

*TABLE I. Labels and their corresponding number.*
B. Lexical parsing

1. Parsing into words

All of the 2400 emails are parsed into words using space character and any of the characters ,!:;'-[]() as delimiters after being converted to lowercase text. The lines of emails are trimmed for empty spaces at the beginning and at the end. For each email, except for the subject line, any line above the line starting with x-filename: is ignored to ignore email header information. Any line starting with — is also ignored to ignore the separator in replies and forwards. Any word containing @ is also ignored to ignore email addresses. After which dot is added to the list of delimiters to account for the end of sentences but not the dot in email addresses. As a result 19401 words are selected which in this paper after being sorted lexicographically are called the initial parsed words and denoted by \( W = (w_1, w_2, ..., w_{19401}) \).

2. Counting the words and calculating normalized frequencies

For each word \( w_i \in W \) the number of emails containing that word is counted. Also for each word \( w_i \) in each email \( e_j \), the number of occurrences of that word in that email is counted and is normalized by dividing it over aggregate sum of occurrences of all words of \( W \) in that email which the result is referred to as normalized per email word frequency in this paper and denoted by \( T_{ji} \).

3. Filtering stop words

The list of lowercase stop words in Table I are used to filter the list of words in \( W \) into a filtered list of words \( \hat{W} \).
| Category 1 | i, we, you, they, she, he, it, this, that, these, those |
| Category 2 | my, our, your, their, her, his, its |
| Category 3 | me, us, you, them, her, him |
| Category 4 | was, were, am, are, is, be, been, being, will, would, could, can, had, has, have, may, might, should |
| Category 5 | and, or, but, also, however, so, because, not, if, then |
| Category 6 | a, an, the |
| Category 7 | what, who, which, where, when, how, did, do, does, why |
| Category 8 | only, all, just, any, few, some, other, much, very |
| Category 9 | one, two, three, four, five, six, seven, eight, nine, ten |
| Category 10 | get, let, want, like |
| Category 11 | here, there, in, of, for, at, as, with, by, on |
| Category 12 | http, https, www, com |
| Category 13 | to, from, subject, cc, bcc, re, fw, forwarded, sent, am, pm, attached, regards, best, find, email, following, thanks, thank, dear, hi, hello, fax, phone, address, e-mail, below, fyi |
| Category 14 | eol |
| Category 15 | monday, tuesday, wednesday, thursday, friday, saturday, sunday |
| Category 16 | january, february, march, april, may, june, july, august, september, october, november, december |

**TABLE II. Stop words (lowercase)**

$\hat{W}$ has 19256 words in it.

4. **Lexical filter**

From $\hat{W}$, any single character word is filtered in addition to any word containing any digit or hyphen or @ or & symbol. The result is called the basic filtered words in this paper and denoted by $\tilde{W}$. The number of basic filtered words is 13688.
C. Finding the top indicative words for each email

1. Calculate per email significance for each of basic filtered words

Term Frequency–Inverse Document Frequency \([2, 3]\) is computed for each of basic filtered words per email and the result is called per email significance for each of basic filtered words in this paper and denoted by \(S_{ji}\) for each email \(e_j \in E\) and each \(\tilde{w}_i \in \tilde{w}_i\). Mathematically speaking:

\[
S_{ji} = \frac{F_{ji}}{\sum_t F_{at}} \cdot \log\left(\frac{N}{\hat{F}_i}\right)
\]

where \(F_{ji}\) is the number of times that the word \(\tilde{w}_i\) appears in the email \(e_j\), and \(\hat{F}_i\) specifies the number of emails that contain the word \(\tilde{w}_i\) at least once, and \(N\) is the number of selected emails (\(N = 2400\)).

2. Calculate per email significance rank for each of basic filtered words

Based on the per email significance for each of basic filtered words, the ranking of each of the basic filtered words in each of the emails is calculated.

3. Selecting first frequency filtered words

Out of 13688 basic filtered words only 2413 appear in at least 10 of the selected emails which those words are called the first frequency filtered words in this paper and denoted by \(H\).

4. Selecting second frequency filtered words

Out of 2413 first frequency filtered words, only 427 appear in the top 100 words in at least 51 emails ranked based on the ranking of basic filtered words for that email and those 427 words after being lexicographically sorted are called second frequency filtered words in this paper and denoted by \(\tilde{H} = (\tilde{h}_1, \tilde{h}_2, ..., \tilde{h}_{427})\).
5. **Extracting per email significance and significance ranks for each of second frequency filtered words**

At this point, the focus is shifted from basic filtered words to second frequency filtered words, so only per email significance of those words are kept but the ranking per email is set to -1 if the word does not appear in the top 100 words for that email based on the ranking of the basic filtered words.

**D. Computing Cosine tailored features per email**

At this stage, the average number of times a word in the second frequency filtered words has appeared in the top 100 words of an email is computed and truncated to the integer part of that number. Then two times that average number is selected as the desired number of ones in the final features per email. Now out of 427 second frequency filtered words, when they appear in the top 100 list for an email (except if the ranking is -1), the feature for that word for that email is considered one. The issue however is that the number of ones is not fixed for each email as desired, so for each email, enough ones are added to make the number of ones equal for all emails but a dimensions used for a dummy 1 for an email cannot be used for a dummy 1 for another email, which leads to significant number of features. The total number of features per email at the end is 48557.

**E. Scrambling the sorting of selected emails**

At the end, before creating the final data set and before evaluation, the order of emails is scrambled using the pseudo-random number generator of go programming language (rand.New and rand.Rand.shuffle functions) initialized with a seed value. Although not perfect, the pseudo random number generator should be practically good enough for removing any tangible bias based on the order of emails in most or all cases (The author cannot imagine a case where more sophisticated random number generators such as the CSPRNG (Cryptographically Secure Pseudorandom Number Generator) type of random generators are needed for this task but the extent of his imagination does not necessarily cover all possible cases).
TABLE III. average number of non zero primary features per email per label number (rounded to 3 decimal places) and standard deviation of number of non zero primary features per email per label number (rounded to 3 decimal places).

III. EVALUATION AND STATISTICS

The evaluation is based on the accuracy of a $k$ nearest neighbours majority voting classification using Cosine distance. In addition to KNN majority voting classification accuracy and confusion matrix, some statistics for the process is reported. The KNN majority voting classification accuracy using Cosine distance with $k = 10$ is 76.75% which shows at least some level of success given the 8 labels involved. The result of conversion is 48557 features per selected email out of which exactly 40 features per email are non-zero. Average number of non zero primary features per email is 19.946 (rounded to 3 decimal places) . Standard deviation of number of non zero primary features per email: 10.963 (rounded to 3 decimal places). The table III shows average number of non zero primary features per email per label number (rounded to 3 decimal places) and standard deviation of number of non zero primary features per email per label number (rounded to 3 decimal places).

The table IV shows the confusion matrix for the KNN majority voting classification.

The table V shows some of the statistics related to the process of creating features.
TABLE IV. Confusion matrix (rounded to two decimal places)

|   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  |
|---|----|----|----|----|----|----|----|----|
| 0 | 6.46% | 0.12% | 1.83% | 2.50% | 0.00% | 0.08% | 1.17% | 0.33% |
| 1 | 0.75% | 9.12% | 0.21% | 0.58% | 0.17% | 0.00% | 0.25% | 1.42% |
| 2 | 0.17% | 0.00% | 10.79% | 1.21% | 0.00% | 0.00% | 0.33% | 0.00% |
| 3 | 0.33% | 0.00% | 1.38% | 10.67% | 0.00% | 0.00% | 0.12% | 0.00% |
| 4 | 0.46% | 0.29% | 0.25% | 0.25% | 10.92% | 0.00% | 0.21% | 0.12% |
| 5 | 0.00% | 0.00% | 0.00% | 0.08% | 0.00% | 12.42% | 0.00% | 0.00% |
| 6 | 1.58% | 0.12% | 3.62% | 2.12% | 0.00% | 0.00% | 4.96% | 0.04% |
| 7 | 0.17% | 0.54% | 0.00% | 0.33% | 0.04% | 0.00% | 0.00% | 11.42% |

IV. FINAL DATA SET

The result of conversion is a data set named MeeeFTCD (Massive Enhanced Extracted Email Features Tailored for Cosine Distance) which is available at [https://github.com/farshad-barahimi-academic-codes/data_sets_preparation/blob/main/data_sets/meeeftcd_data_set.zip](https://github.com/farshad-barahimi-academic-codes/data_sets_preparation/blob/main/data_sets/meeeftcd_data_set.zip)
and  
https://web.cs.dal.ca/~barahimi/data-sets/meeeftcd/  
The code used to create the data set and the results of this paper can be found in  
https://github.com/farshad-barahimi-academic-codes/data_sets_preparation  
for which a guide is available at  
https://web.cs.dal.ca/~barahimi/data-sets/meeeftcd/how_was_built.  
which relies on compiling the code written in go programming language and passing right  
arguments to it.

[1] Enron email dataset,  
https://www.cs.cmu.edu/~enron/  
date accessed: April 24, 2022; Version: May 7, 2015; Posted by William W. Cohen.

[2] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, The influence of preprocessing on text classification using a bag-of-words representation,  
PLOS ONE 15, 1 (2020).

[3] S. Aryal, K. M. Ting, G. Haffari, and T. Washio, Beyond tf-idf and cosine distance in documents dissimilarity measure, in Information Retrieval Technology, edited by G. Zuccon, S. Geva, H. Joho, F. Scholer, A. Sun, and P. Zhang (Springer International Publishing, Cham, 2015) pp. 400–406.