Journal of Computer Science 10 (4): 689-696, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.689.696 Published Online 10 (4) 2014 (http://www.thescipub.com/jcs.toc)

A NOVEL EMAIL RESPONSE ALGORITHM FOR EMAIL MANAGEMENT SYSTEMS

Abdulkareem Al-Alwani

Yanbu University College Yanbu, Saudi Arabia

Received 2013-11-20; Revised 2013-11-30; Accepted 2013-12-23

ABSTRACT

Email has been one of the most commonly used tool for communication in the recent years and email management has evolved as a major challenge due to prevailing situation of online email congestion. This study presents a novel algorithm for automatic email response methodology in an Email Management System to minimize email overload. The proposed model uses Bayes classifier to categorize emails into classes and generate suitable replies to these classes using information extraction and template filling. Our research aims to intelligently automate email response using Naïve Bayesian classification and formulate probabilistic dictionaries for accurate information extraction. This research will help in reducing email overload and unavoidable congestion by employing a novel email response architecture for an email management systems.

Keywords: Email Messages, Naïve Bayes, Email Classification, Information Extraction, Email Template, Email Reply, Unsupervised Learning, Email Management System

1. INTRODUCTION

Email is one of the most reliable means of online correspondence and has become an essential communication tool for most organizations and individuals. With the increase in email usage, prioritization and organization of emails becomes an overwhelming challenge. An average user spends a considerable amount of time in reading, understanding and responding to emails. Furthermore, most of the emails follows a fixed structure in terms of content and require simple replies e.g., recent study showed an email reply system for a company's Frequently Asked Questions (FAQ) queries. In FAQ, most of the questions are repeated by multiple users and using proper classification, correct replies can be generated to answer the user queries (Kosseim *et al*., 2001). To assist users in automated email replies, with correct classification and timely prioritization, we present a novel algorithm for generating automated priority reply to emails after appropriate classification. This algorithm will have the provision for accurate email reply prediction employing unsupervised learning subroutines. The proposed algorithm will provide concise, highly structured and prioritized emails. It will also help develop a framework for email reply system capable of generating automatic replies to incoming emails. This will save the effort and time wasted when browsing through each email one by one, in turn, assisting the user with email management in an efficient manner. The algorithm model is based on Naïve Bayer model for email classification and Markov probabilistic methods for information extraction to facilitate template filing. Naïve Bayes is a popular method, a frequently used machine learning model for several years. Its simplicity allow it to be used easily manner in many applications and good classification results are obtained using this learning approach despite its dependence on an unrealistic independence assumptions. For this reason, much research has been published on Naive Bayes classification approaches in real world applications (McCallum and Nigam, 2003; Rish, 2001; Rish *et al*., 2001; Swezey *et al*., 2012; Hossain *et al*., 2013). For information extraction and subsequent template matching, string matching (Al-mazroi and Rashid, 2011) and probabilistic methods (Khatatneh *et al*., 2006) are employed. Markov

probability model is used to train the email system on previously observed patterns to facilitate appropriate template selection for email reply. Overall, this in this research we have proposed a novel algorithm to facilitate email management systems by generating automated intelligent replies to selected class of emails.

Next section discusses literature pertaining to our area of research. A detailed description of the proposed algorithm is presented in section 3 followed by conclusion with closing remarks on future course of work.

2. RELATED WORK

Email is one of the most reliable methods for online communication and correspondence. While reviewing important literature, main focus was kept on research work related to email interaction methodologies along with the core techniques that can be used for an efficient email response system. Regarding user's familiarity with emails, there are five main activities encompassing user's interaction with an email system. These activities are Flow, Triage, Task management, Archive and Retrieve (Cadiz *et al*., 2001) and are vital to understanding workflow of an email process. User interaction with an email system act as a basis for classification, as automated response must follow a similar interaction routine in the algorithm to generate a human like reply. Mackay (1988) showed that users handle email in a multitude of ways. He also emphasized that email users can be classified in two main categories based on their interaction with emails, which are:

- Prioritizers
- Archivers

Prioritizers tend to keep the email inflow and outflow in check, keeping tight control of their email database, whereas archivers save the information for later use to prevent missing important emails. Initial categorization of user classes helps in analyzing the email interactions unique to both classes.

A user cannot handle typical email overload and congestion as any inaccurate classification by user can lead to loss of important information. Steve and Sidner (1996) reported that in addition to normal communication, the email system is congested and overloaded f or its applicability in a variety of tasks such as professional correspondence, reminders, task and contact management and information backup. The issue of email congestion was investigates in a study by Laclavik and Maynard (2009). The study showed that an average email user sends and receive emails every day to reach a point of overload and information congestion. They proposed semantic web based approach for managing email congestion, but the technique required manual annotations for initial classification. In another research (Beseiso *et al*., 2012), ontology based architecture was employed to handle unstructured emails in a symantic database. This research work presented ontology learning and extraction process effectiveness in keeping right track of important information in an event of email overload.

As email is basically a collection of electronic text based words, machine learning methods can be used with superior performance for classification of electronic documents. Yang and Kwok (2012) showed that machine learning techniques can be used to intelligently classify emails in multiple categories. A detailed study was published in 2009, which discussed application of artificial intelligence techniques in intelligent email architecture. In this study authors identified improvements to user-email interface and used machine learning routines to support these changes in real-time targeting issues like reply prediction, attachment prediction and summary keyword generation (Dredze and Wallach, 2009). In a review study on natural language processing techniques by Jackson *et al*. (2012) authors researched whether natural language processing techniques can be used to fully automate the extraction of knowledge from emails. This study reports four generations of building systems to share knowledge and discusses efficacy of knowledge extractions for these four generations (Jackson *et al*., 2012). Machine Learning techniques, Data Mining and Natural Language Processing (NLP) work in combination to automatically identify patterns from the electronic documents to help classify them in intended categories (ALmomani *et al*., 2012). Naïve bayes classifier was found to be most effective in real world complex scenarios due to simple initial conditions required by the model (Baharudin *et al*., 2010). Naïve Bayes classifiers can be trained in an efficient manner. There performance is characterized by the nature of the applied probability model. A small training dataset is sufficient to estimate required statistics which are necessary for accurate classification and categorization.

Information extraction and appropriate template assignment must be robust enough to handle a wide range of textual data using efficient probabilistic processes. These two processes are imperative in generating an accurate email response to a class of email. In related work, Gwizdka (2001) showed that adopting human like prospective memory into the algorithm can help

make better response decisions based on past decision states. The prospective memory includes any previous information parameter that can be used to determine a better reply decision. This study underlines the viability of probabilistic methods in characterizing an email response. In another probabilistic method, Ayodele *et al*. (2011) used Email Urgency Reply Prediction (EURP) model to prioritize emails that require urgent response. In a related research, authors analyzed various text formats and generating a customized and linguistically-motivated answer to emails related to frequently asked questions using information extraction and intelligent template filling (Kosseim *et al*., 2001).

Our proposed algorithm is formulated with a view to provide an integrated email classification and template matching methodology to sustain an intelligent email response system. A detailed discussion on algorithm architecture is presented in the next section.

3. PROPOSED ALGORITHM

The primary aim the proposed algorithm is to generate intelligent automatic replies to selected emails based on their content. The email types that we want to be able to respond will thus from now be called 'classes' of emails. For each class we will have the 'template' which will further be filled in with the information extracted from the email. The algorithm for automatic email reply consists of two parts:

- Email classification
- Information extraction and template filling

3.1. Email Classification

As previously mentioned this research is aimed to develop algorithms which will able to automate responses to predefined classes of emails e.g., meeting, product support, customer support, product order. The first stage of the algorithm is to classify the incoming email into one of such classes. The classification is carried out using Naive Bayes with Laplacian Smoothing technique. In order to classify emails we will first have to build dictionary from existing emails and train Naive Bayes using dictionary in order to formulate primary probability parameters.

Detailed discussion on the classification approach used in this algorithm to categorize emails is presented in the following subsections.

3.2. Email Structure

The email content is structured in the following manner. First, the email will be separated into a title and a body. After that for each part we will use the Bag of Words representation. That means that words in each bag or collection will be sorted irrespective their order. We will assume that the order of the words is irrelevant. This assumption may seem weak but it has given good results in text classification using Naïve Bayes classification (Frank and Bouckaert, 2006). This will be done for both title and body.

3.3. Email Class

As mentioned above we will calculate the email class based on the words in the title and in the body text. This is carried out using Naive Bayes model (Frank and Bouckaert, 2006). We denote number of words in the title 'n' and number of words in the body by 'm'. Let C_i be the i^{th} class of the email. Title words are denoted as 'TW' and body words as 'BW'. For each predefined email class C_i, we will calculate conditional probability Equation (1):

$$P(C_i \mid TW_1, TW_2, ..., TW_n, ...BW_1, BW_2, ...BW_m) \quad (1)$$

We will then classify the email in the class with the highest probability. Probability in (1) iscalculated using following Bayesian formula Equation (2):

$$\frac{P(TW_1\, TW_2 ...TW_n BW_1\, BW_2 ...BW_m \mid C_i)P(C_i)}{P(TW_1\, TW_2 ...TW_n BW_1\, BW_2 ...BW_m)} \quad (2)$$

The words in our model are independent of the class. The upper term is calculated with ease via the product rule. The lower term is normalizing constant and it is calculated as the sum of the upper terms.

Since we only need the maximum probability, we need only calculate the unnormalized probability for each class. This is carried out by calculating required probabilities using following Equation (3):

$$\begin{aligned}&\bar{P}(C_i \mid TW_1\, TW_2 ...TW_n BW_1\, BW_2 ...BW_m)\\ &= P(C_i)\prod_{j=1}^{n} P(TW_j \mid C_i)\prod_{j=1}^{m} P(BW_j \mid C_i)\end{aligned} \quad (3)$$

We will then assign the class with the highest unnormalized value. The only thing left to do is to learn the following parameters of the model:

- $P(C_i)$ for each class C_i
- $P(TW_j|C_i)$ for each word in the title and each class
- $P(BW_j|C_j)$ for each word in the body and each class

To elaborate the classification process, we present an example. Suppose we have 3 classes meeting, product support, ordering. With each class probability is assigned p_i, where i = (1,2,….,n) representing i^{th} class. Assuming initial probabilities are $p_1 = 0.4$,$p_2 = 0.35$ and $p_3 = 0.25$.

Consider following email:

Title: Meeting schedule
Body:
Hi,
I would like to ask you if you are available for a meeting at 10.00 am tomorrow.
Best Regards

The classification proceeds as follows:

Step1: Initialze p1, p2 and p3 to initial probabilities of the classes according to the training data.

Step2: For each word in title read its value for each of the class and multiply it Lets say we have dictionary for email title, we calculate using **Table 1**.
p1 = 0.4*0.3*0.2 = 0.024
p2 = 0.35*0.001*0.01 = 0.0000035
p3 = 0.25*0.01*0.07 = 0.000175

Step3: Exactly the same procedure but we now use the dictionary for the email body and multiply the values for each word in the body.

3.4. Parameter Learning

Parameter learning is carried out using maximum likelihood technique with Laplacian Smoothing (Hansen and Johnson, 2005), keeping k = 1. For accurate parameter learning, we will need a large set of training data. This set could be composed of previous emails. We will use this emails to train our model for effective parameter learning.

3.5. Learning of P(Ci) Parameters

Suppose, we have a training set with 'N' emails and that we have K_i emails for each class i. Lets also assume that we have M classes Since we are using maximum likelihood with Laplacian smoothing with k = 1 then parameters is calculated in the following way Equation (4):

$$\bar{P}(C_i)\frac{K_i + 1}{K + M} \quad (4)$$

3.6. Dictionary Learning

An effective approach for learning $P(TW_j|C_i)$ and $P(BW_j|C_i)$ parameters is by creating a dictionary. Two dictionaries are created, one each for email body and title. As initially stated, there are M classes of emails. Dictionary D will be a matrix of W×M elements, where W is the total number of words. Following routine will be followed in building a dictionary using an email. Iterate through all the words in the email title/body. For each word 'w:

- Check if the word is already in the dictionary
- If the word is in dictionary go to step(vi)
- Otherwise create new row in the dictionary associating it with word w
- Initialize each of the M fields (one for every email class) with k = 1 (Laplacian smoothing parameter)
- Go to step(vii)
- Find the row which is associated with the word 'w'
- Find the class i of the email from which is word 'w'
- Increment the value of the cell D_{wi}

After the dictionaries are formulated, required probabilities are calculated using these dictionaries. We designate dictionary for the title as 'DT' and dictionary for the body as 'DB'. Then respective probabilities are given as Equation (5 and 6):

$$P(TW_j \mid C_i) = DT_{ji} \Big/ \sum_{k=1}^{W} DT_{ki} \quad (5)$$

$$P(BW_j \mid C_i) = DB_{ji} \Big/ \sum_{k=1}^{W} DB_{ki} \quad (6)$$

After probabilities are calculated from (5) and (6), emails are categorized into classes based on the calculated probabilities.

Suitable replies are then generated using information extraction and template filling, which is discussed in the following section.

3.7. Information Extraction and Template Filling

After we have classified the incoming email into one of the predefined classes we should generate appropriate email response. Since we already have predefined classes we can assume that the responses for the same class will look similar. Therefore each class will have some predesigned template (one or more). Each template will then be filled with the relevant information extracted from the email.

3.8. Template Generation and Selection

The templates will be created manually with blanks left to be filled in with relevant information from the

email. For example template for meeting schedule could look something like:

Dear————,
We are confirming that the meeting will take place at ____ on _____.
Best Regards.

The blanks will be filled with name of the sender, event time and date of the meeting.

Selection of the template can be categorized in two cases:

- When the email class has one template
- When the email class could have more than one templates

First case is easy to address since there is only one template to use. In the case when we could have more than one template, template choice can be made using two ways:

- Static
- Dynamic

The static category is used when reply depends only on the email received. In this case, the class with multiple templates will be further categorized into subclasses, one for each template. In that case the email will be classified into one of the subclasses followed by selection of an appropriate template. For dynamic template selection, the decision does not depend only on the text of the received email but also on the state of our system. In this case, relevant information must be extracted first and an appropriate template will be selected using these parameters. The required parameters needed from the email will be called decision variables. Detailed discussion information extraction will be presented in the subsequent sections. An example for dynamic template selection could be meeting schedule where the information will be time and date of the meeting. We will then check in our relevant local system state i.e., our calendar. If a slot for requested time and date is available, an appropriate response (positive or negative one) is selected and sent as an email reply. Basic steps followed in dynamic selection process are summarized as follows:

- Extract the decision variable from the email
- Pass the decision variable into the system in order to get the response
- Select the appropriate template based on response

3.9. Information Extraction

Selection of template is largely dependent upon the information extracted. We will now explain the extraction of the relevant information from the email. We segregate the content of an email as two information types:

- Decision variables information
- Template information

Both types are further elaborated in the following subsections.

3.10. Decision Variables

Each class with dynamic template selection will have its decision variable e.g., date and time for meeting, GPA for scholarship. In this case, we will assume that the process is Markov and that the probability that word is decision variable depends on the word itself along with the word before and after it. For each word in the email we will calculate the probability that that word represents the decision variable. After that we will use the word with the highest probability. Following relation is used to determine that a word w_i is a decision variable Equation (7):

$$P_b(w_{i-1})P_d(w_i)P_a(w_{i+1}) \quad (7)$$

Where:
$P_b(w)$ = The probability that word w is word before decision variable
$P_d(w)$ = The probability that word w is decision variable
$P_a(w)$ = The probability that word w is word after decision variable

P_b and P_a will be read from the dictionary while for P_d, probabilistic template matching will be used.

3.11. Template Information

For each template we should extract information for each blank that template has. Same as in decision variable extraction, we also assume that this is a Markov process. Assuming, we have 'K' blanks in the template. Information type is associated with every blank (name, date, product name). For each blank we will check each word 'w_i' from the receiving email and calculate the probability:

$$P_b(w_{i-1})P_i(w_i)P_a(w_{i+1}) \quad (8)$$

Where:

$P_b(w)$ = The probability that word w is word before desired information
$P_i(w)$ = The probability that word w is the desired information
$P_a(w)$ = The probability that word w is word after desired information

As before P_b and P_a will be read from the dictionary while for P_i probabilistic template matching will be used.

For example, we want to extract information from the first email whose dictionary is given in **Table 1**. We want to extract the time of the meeting. We will calculate Equation (8) for each word in the email.

Hi,
I would like to ask you if you are available for a meeting at 10.00 am tomorrow.
Best Regards

We start with the word 'Hi'. The word before the Hi is the empty word and the word after is I. As we said we have two dictionaries here D_A and D_B. We read the value of the empty word from D_B and the value of the word I form the D_A and multiply them. Suppose we have D_B("") = 0.002 and D_A ("I") = 0.001. We then calculate the edit distance from the word Hi form the time template. Taking edit distance as 5. We calculate P_d as 1/(1+5) = 1/6. We now multiply these 3 values we get:

$$0.002 * 0.001 * 1 / 6 = 3.33 * 10 - 7$$

We now repeat the process for every word. We take the word with highest probability to be the time of the meeting.

3.12. Dictionary Building

Based on our classes and templates, we should build two dictionaries for each decision variable and for each information type. One dictionary will be for a word before and one dictionary for the word after. Following routine is followed for building these dictionaries:

i. Manually mark decision variables and relevant information in the emails which represent the training set
ii. For each decision variable, 'd'
  iii. Create two empty dictionaries DA and DB
  iv. For each word w from the training set emails
    v. Check if the word is already in the dictionary
    vi. If not go to step (ix)
    vii. Add the word to both dictionaries and initialize the value to
    viii. Go to step (x)
    ix. Retrieve the row of the word
    x. Check if the word after w is labeled as decision variable d
    xi. If yes increment the entry in DB
    xii. Check if the word before w is labeled as decision variable
    xiii. If yes increment the entry in DA
  xiv. Normalize D_A and D_B so they sum up to 1
xv. For each relevant information, 'r'
xvi. Create two empty dictionaries DA and DB
  xvii. For each word w from the training set emails
    xviii. Check if the word is already in the dictionary
    xix. if not go to step 9
    xx. add the word to both dictionaries and initialize the value to
    xxi. go to step 10
    xxii. retrieve the row of the word
    xxiii. Check if the word after w is labeled as relevant information
    xxiv. If yes increment the entry in D_B
    xxv. Check if the word before w is labeled as relevant information
    xxvi. If yes increment the entry in D_A
xxvii. Normalize D_A and D_B so they sum up to 1

After the dictionaries are constructed, they can be employed to calculate the appropriate probabilities for information extraction and template filling. Following is an example of a small dictionary constructed for information extraction. Let's say we want to extract date, product name, name of the customer.

Examples of built dictionaries are shown below. **Table 2** shows a dictionary used for classification of emails and **Table 3** contains dictionary for information extraction. **Table 3** contains probabilities of the words succeeded by information of interest and a similar structure is followed for word after the information.

Table 1. Dictionary probability table for classification

	Meeting	Product support	Ordering
Meeting	0.3	0.001	0.01
Schedule	0.2	0.010	0.07

Table 2. Dictionary example for classification

Word/class	Meeting	Support	Ordering
At	0.300	0.10	0.20
Available	0.400	0.05	0.10
Order	0.001	0.05	0.50
Help	0.050	0.40	0.01
Hi	0.100	0.20	0.10

Table 3. Dictionary example for information extraction

Word/information	Date	Product name	Name of the customer
At	0.400	0.0100	0.001
Dear	0.001	0.0001	0.400
The	0.001	0.3000	0.050
In	0.200	0.1000	0.001
Of	0.050	0.2000	0.001

3.13. Probabilistic Template Matching

We will now explain probabilistic template matching for calculation of P_i and P_d. The algorithm is the same for both cases. Each set of information will have a relevant template, list of templates or a list of possible values e.g., Time will have templates like HH:MM, HH:MM: S. Months can have possible values (January, February). Names can have both values and templates (List of names, [A-Z][a-z]). List of values and templates are created for each decision variable and relevant information. The calculation of P_d is carried out as follows:

- For each word ‘w’ in the email
- Set the Pd to 0
- For each element ‘e’ in the template_value list
- Calculate the minimum edit distance d from ‘w’ to ‘e’
- Set p = 1/(1+d)
- If p >P_d set P_d to p

The procedure followed here will be used again for calculation of P_i. Now the values of P_d/P_i can be used together with values from the dictionaries mentioned above to extract relevant information.

4. CONCLUSION

In this research, we have proposed an algorithm for generating intelligent replies to certain classes of emails. This algorithm is intended to build a framework for email management system, capable of classifying and automatically replying to a certain class of emails. The algorithm performs classification of emails based on the content with the help of information extraction. Template based reply is generated, customized to cater for the requirements of email at hand. In order to increase the domain of application of our algorithm, new knowledge-bases, extraction methodologies and smart templates may be developed.

The quality of email content analysis is a pivotal parameter that is critical for intelligent e-mail answering, as a wrong email reply to a client can lead unintended situations. So, a step by step approach to an independent email answering system is necessary to perfect email answering technology; i.e., testing viability of the system on predetermined classes of email and eventually increasing the class number and relevant templates after careful training of the intended system.

The next step in this research is to develop an email management system to successfully implement the proposed algorithm for real time generation of email responses. As the intended framework is under development, no strict evaluation has been performed yet. As far as further research is concerned, our main aim is to implement this model in a suitable email application to test its efficacy by measuring performance parameters e.g., Response accuracy, precise content filling, template relevance of an efficient email response system.

5. REFERENCES

Al-mazroi, A.A. and N.A. Rashid, 2011. A fast hybrid algorithm for the exact string matching problem. Am. J. Eng. Applied Sci., 4: 102-107. DOI: 10.3844/ajeassp.2011.102.107

ALmomani, A., T.C. Wan, A. Altaher, A. Manasrah and E. ALmomani *et al*., 2012. Evolving fuzzy neural network for phishing emails detection. J. Comput. Sci., 8: 1099-1107. DOI: 10.3844/jcssp.2012.1099.1107

Ayodele, T., C.A. Shoniregun and S. Zhou, 2011. Email urgency reply prediction. Proceedigns of the International Conference on Information Society (i-Society), Jun. 27-29, IEEE Xplore Press, London, pp: 418-422.

Baharudin, B., L. Lee and K. Khan, 2010. A review of machine learning algorithms for text-documents classification. J. Adv. Inform. Tech., 1: 4-20. DOI: 10.4304/jait.1.1.4-20

Beseiso, M., A.R. Ahmad and R. Ismail, 2012. A new architecture for email knowledge extraction. Int. J. Web Semantic Tech., 3: 1-10.

Cadiz, J.J., L. Dabbish, A. Gupta and G.D. Venolia, 2001. Supporting email workflow. Microsoft Res.

Dredze, M. and H. Wallach, 2009. Intelligent email: Aiding users with AI. Selected Works of Hanna M. Wallach.

Frank, E. and R. Bouckaert, 2006. Naive bayes for text classification with unbalanced classes. Proceedings of the 10th European conference on Principle and Practice of Knowledge Discovery in Databases, Sept. 18-22, Springer-Verlag Berlin, pp: 503-510. DOI: 10.1007/11871637_49

Gwizdka, J., 2001. Supporting prospective information in email. Proceedings of the Extended Abstracts on Human Factors in Computing Systems, Mar. 31-31, ACM Press, New York, 135-136. DOI: 10.1145/634067.634150

Hansen, C.D. and C.R. Johnson, 2005. The Visualization Handbook. 1st Edn., Academic Press, Burlington, MA., ISBN-10: 012387582X, pp: 962.

Hossain, J., N. FazlidaMohdSani, A. Mustapha and L. SurianiAffendey, 2013. Using feature selection as accuracy benchmarking in clinical data mining. J. Comput. Sci., 9: 883-888. DOI: 10.3844/jcssp.2013.883.888

Jackson, W.T., S. Tedmori and H. Hinde, 2012. The boundaries of natural language processing techniques in extracting knowledge from emails. J. Emerg. Technol. Web Intell., 4: 119-119.

Khatatneh, K., I.M.M. El-Emary and B.A. Rifai, 2006. Probabilistic artificial neural network for recognizing the arabic hand written characters. J. Comput. Sci., 2: 879-884. DOI: 10.3844/jcssp.2006.879.884

Kosseim, L., S. Beauregard and G. Lapalme, 2001. Using Information Extraction and Natural Language Generation to Answer E-Mail. In: Natural Language Processing and Information Systems, Bouzeghoub, M., Z. Kedad and E. Metais (Eds.), Springer, Berlin, ISBN-10: 3540419438, pp: 152-163.

Laclavik, M. and D. Maynard, 2009. Motivating intelligent e-mail in business: An investigation into current trends for e-mail processing and communication research. Proceedings of the IEEE Conference on Commerce and Enterprise Computing, (EC' 09), IEEE Xplore Press, Vienna, pp: 476-482. DOI: 10.1109/CEC.2009.47

Mackay, W., 1988. Diversity in the use of electronic mail: A preliminary inquiry. ACM Trans. Office Inform. Syst., 6: 380-397. DOI: 10.1145/58566.58567

McCallum, A. and K. Nigam, 2003. A comparison of event models for naïve bayes text classification. J. Mach. Learn. Res.

Rish, I., 2001. An empirical study of the naïve bayes classifier. IBM Research Division.

Rish, I., J. Hellerstein and J. Thathachar, 2001. An analysis of data characteristics that affect naïve bayes performance. IBM T.J. Watson Research Center.

Steve, W. and C. Sidner, 1996. Email overload: Exploring personal information management. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Apr. 13-18, New York, pp: 276-283. DOI: 10.1145/238386.238530

Swezey, R.M.E., S. Shiramatsu, O. Tadachika and S. Toramatsu, 2012. An Improvement for Naive Bayes Text Classification Applied to Online Imbalanced Crowdsourced Corpuses. In: Modern Advances in Intelligent Systems and Tools, Ding, W., H. Jiang, M. Ali and M. Li (Eds.), Springer, London, ISBN-10: 3642307310, pp: 147-152.

Yang, W. and L. Kwok, 2012. Improving the automatic email responding system for computer manufacturers via machine learning. Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering (ICIII), Oct. 20-21, IEEE Xplore Press, Sanya, pp: 487-491. DOI: 10.1109/ICIII.2012.6340024