Automatic email response suggestion for support departments within a university

Aditya Parameswaran [1], **Dibyendu Mishra** [1], **Sanchit Bansal** [1], **Vinayak Agarwal** [1], **Anjali Goyal** [1], **Ashish Sureka** [Corresp. 1]

[1] Computer Science, Ashoka University, Sonepat, Haryana, India

Corresponding Author: Ashish Sureka
Email address: ashish.sureka@ashoka.edu.in

Background. Office of Academic Affairs (OAA), Office of Student Life (OSL) and Information Technology Helpdesk (ITD) are support functions within a university which receives hundreds of email messages on the daily basis. A large percentage of emails received by these departments are frequent and commonly used queries or request for information. Responding to every query by manually typing is a tedious and time consuming task and an automated approach for email response suggestion can save lot of time.

Methods. We propose an application and solution approach for automatically generating and suggesting short email responses to support queries in a university environment. Our proposed solution can be used as one tap or one click solution for responding to various types of queries raised by faculty members and students in a university. We create a dataset for the application domain and make it publicly available. We apply a machine learning framework for classifying emails into categories such as office of academic affairs or information technology department. We apply a machine learning based classification approach for sub-category level classification also. We apply text pre-processing techniques, feature selection, support vector machine and naïve naive classifiers. We present an approach to overcome various natural language processing based challenges in the text.

Results. We conduct a series of experiments and evaluate the approach using confusion matrix and accuracy based metrics. We study the discriminatory power of features and compare their relevance for the classification task. Our experimental results reveal that the proposed approach is effective. We conclude from our experiments that discriminatory features can be extracted from the text within our specific domain and automatic email response suggestion can be accurately created using machine learning algorithms and framework. We experiment with two different learning algorithms and observe that SVM outperforms Naïve Bayes. We achieve a classification accuracy of above 85% for all the classes and sub-classes.

Discussion. Our experiments on email response suggestion are conducted on a corpus consists of short and frequent emails by a university function but the proposed approach and techniques can be generalized to other domains also. We observe that different classifiers give different results and there is a significant difference in the predictive power of features.

Automatic Email Response Suggestion for Support Departments within a University

Aditya Parameswaran[1], Dibyendu Mishra[1], Sanchit Bansal[1], Vinayak Agarwal[1], Anjali Goyal[1], and Ashish Sureka[1]

[1]**Ashoka University, Haryana, India**

Corresponding author:
Ashish Sureka[1]

Email address: ashish.sureka@ashoka.edu.in

ABSTRACT

Background. Office of Academic Affairs (OAA), Office of Student Life (OSL) and Information Technology Helpdesk (ITD) are support functions within a university which receives hundreds of email messages on the daily basis. A large percentage of emails received by these departments are frequent and commonly used queries or request for information. Responding to every query by manually typing is a tedious and time consuming task and an automated approach for email response suggestion can save lot of time.
Methods. We propose an application and solution approach for automatically generating and suggesting short email responses to support queries in a university environment. Our proposed solution can be used as one tap or one click solution for responding to various types of queries raised by faculty members and students in a university. We create a dataset for the application domain and make it publicly available. We apply a machine learning framework for classifying emails into categories such as office of academic affairs or information technology department. We apply a machine learning based classification approach for sub-category level classification also. We apply text pre-processing techniques, feature selection, support vector machine and naïve naive classifiers. We present an approach to overcome various natural language processing based challenges in the text.
Results. We conduct a series of experiments and evaluate the approach using confusion matrix and accuracy based metrics. We study the discriminatory power of features and compare their relevance for the classification task. Our experimental results reveal that the proposed approach is effective. We conclude from our experiments that discriminatory features can be extracted from the text within our specific domain and automatic email response suggestion can be accurately created using machine learning algorithms and framework. We experiment with two different learning algorithms and observe that SVM outperforms Naïve Bayes. We achieve a classification accuracy of above 85% for all the classes and sub-classes.
Discussion. Our experiments on email response suggestion are conducted on a corpus consists of short and frequent emails by a university function but the proposed approach and techniques can be generalized to other domains also. We observe that different classifiers give different results and there is a significant difference in the predictive power of features.

1 INTRODUCTION

1.1 Research Motivation and Aim

Office of Academic Affairs (OAA), Office of Student Life (OSL) and Information Technology Helpdesk (ITD) are support functions within a university which receives hundreds of email messages on the daily basis. Email communication is still the most frequently used mode of communication by these departments. A large percentage of emails received by these departments are frequent and commonly used queries or request for information. The authors of this paper are faculty members, teaching fellow and students from a university[1] and based on our interaction with various support functions in the university, we infer that lot of emails are received by support functions such as OAA, OSL and ITD (sometimes even email overload). Responding to every query by manually typing is a tedious and time consuming task.

[1] https://www.ashoka.edu.in/

Furthermore a large percentage of emails and their responses consists of short messages. For example, an IT support department in our university receives several emails on Wi-Fi not working or someone needing help with a projector or requires an HDMI cable or remote slide changer. Another example is emails from students requesting the office of academic affairs to add and drop courses which they cannot do it directly. Kannan et al. proposed an email response suggestion system integrated in Gmail (Kannan et al., 2016). The solution proposed by Kannan et al. solution approach is general (not specific to any particular domain or context) and addresses a limited types of emails. However, based on our literature survey, we infer that the application of automatic email response suggestion system for specific domains is relatively unexplored. For example, automatic email response suggestion for airline ticket booking domain, complaints regarding products and services of an e-commerce company or support functions within a university. There is no dataset or corpus available for conducting research on a diverse variety of application domains. Our motivation is to investigate the application of automatic email response suggestion system for a university support function domain. Our aim is to create a dataset or corpus for the university support function domain and make it publicly available. Our specific aim is to investigate machine learning based text classification techniques for generate email responses to short messages received by departments like information technology helpdesk, office of academic affairs and office of student life within a university.

1.2 Related Work

Kannan et al. propose a method for automatically generating short email responses which is used in Gmail system (Kannan et al., 2016). Their approach is based on deep learning and long short term memory networks (LSTMs) (Kannan et al., 2016). They also solve the problem of creating the most likely email response for a given message (Kannan et al., 2016). Christophe et al. work on a related problem of proactive recommendation of email attachments (Van Gysel et al., 2017). They conduct their study on an enterprise email corpus and propose a weakly supervised machine learning approach for the task of recommending attachable items to the user (Kannan et al., 2016). Yang et al. conduct a research study on email reply behaviour (Yang et al., 2017). They present an approach on predicting email reply behaviour and describe a method for determining whether a recipient will reply to a given email and the time it will take to reply (Yang et al., 2017). Dotan Di Castro et al. conduct a study on user actions on received messages (Di Castro et al., 2016). They study a large number of Yahoo mail users and study actions like read, reply, delete and delete without read (Di Castro et al., 2016). Graus et al. present a study on recipient recommendation for emailing in enterprises (Graus et al., 2014). Their approach is based on the communication graph as well as the email content (Graus et al., 2014).

Alwani et al. propose probabilistic model using Natural Language Processing for email response generation (Al-Alwani, 2015). The proposed technique first extracts attributes from email message and then assign weights to the extracted attributes. The weighted attributes are then related using probabilistic models to fill the available templates for email replying. Sneiders et al. modelled automatic reply of email messages as text categorization problem (Sneiders et al., 2017). They evaluated performance of text-pattern matching technique by analyzing multiword expressions. The results show text-pattern matching can achieve precision value up to 90%. Henderson et al. propose a feedforward network based email response system and evaluates it on Smart Reply application (Henderson et al., 2017). Rather than using LSTM to compute conditional probability, the proposed model uses feed-forward approach over the response sequence. The results show that usage of feed forward deep networks with n-gram outperforms sequence-to-sequence modeling. Ayodele et al. propose an email reply prediction approach using unsupervised learning (Ayodele et al., 2009). Their approach predicts whether an email message requires reply or not. This prediction is based on presence of important noun phrases, question words or marks and date-time in email message (Ayodele et al., 2009).

1.3 Research Contributions

In context to existing work, the study presented in this paper makes the following novel and unique research contributions.

Novel Application Domain – The study presented in this paper is the first on the application of automatic short message response suggestion in the domain of a university support functions such as office of academic affairs, information technology department and office of student life. While there has

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

been some work done in the area of automatic email response suggestion, its application in diverse domains is relatively unexplored.

Dataset Creation – Annotated real world dataset or dataset which is representative of real-world scenario is required for conducting empirical and data-driven based research. We create the first dataset on the classification problem in our domain and make it publicly available through Figshare (Singh et al., 2018). Our dataset can be used by other researchers for building novel approaches and also comparing with our approach.

Experimental Evaluation – We conduct a series of experiments using various text processing techniques, a feature selection technique, approaches to overcome problems in free-form natural language email text and two different classifiers. We examine the effectiveness of our approach and present our insights and results. We provide an in-depth analysis of the working of the underlying system such as the relative importance of terms and their discriminatory power and study their characteristics. To the best of our knowledge, the study presented in this paper is the first machine learning application results for the specific domain of automatic short message response suggestion in the domain of a university support functions.

2 MATERIALS AND METHOD

2.1 Experimental Dataset

Table 1 presents details about our experimental dataset. We created the experimental dataset ourselves as there is no existing publicly available dataset for the specific problem addressed by us in this work. Our dataset is uploaded to Figshare (Singh et al., 2018) website and publicly available. As shown in Table 1, we create three categories (OAA, ITD and OSL) and 13 sub-categories. Table 1, displays the abbreviation and following is the expansion for the 16 abbreviations.

OAA - Office of Academic Affairs

ITD - IT Department

OSL - Office of Student Life

WFO - WIFI Outage

LOD - Login Details

CLK - Clicker

IDC - ID Card

CLE – Class Room Equipment

DPC - Dropping Course

CTM - Course Timings and Clashes

COF - Course Offered

CDT - Courses and DS Registration Timing ADC - Adding Course

RBK - Room Booking

BCL - Room Booking Cancellation

MSD - Meeting Scheduling

RMB - Reimbursement

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

DPT	SCT	Emails	ABR	SPL	SYN	PLY	0 TC	1 TC	2 TC	3 TC
ITD	WFO	20	3	3	3	1	10	10	0	0
	LOD	21	3	1	2	0	16	4	1	0
	CLK	19	3	4	5	0	10	6	3	0
	IDC	16	16	3	0	0	0	13	3	0
	CLE	19	8	2	4	0	9	7	3	0
OAA	DPC	18	14	4	5	1	3	10	4	3
	CTM	19	12	6	0	0	4	12	3	0
	COF	21	14	5	1	0	6	10	5	0
	CDT	16	11	3	0	0	4	8	3	0
	ADC	18	11	2	2	1	8	7	3	0
OSL	RBK	18	5	0	0	5	8	10	0	0
	BCL	16	7	2	1	1	6	7	2	0
	MSD	19	6	3	4	1	8	8	3	0
	RMB	16	4	4	1	0	9	5	2	0
Total		256	114	42	28	10	101	117	35	3

Table 1. Experimental Dataset Details

As shown in Table 1, we create 256 emails covering all the categories and sub-categories. We create 95 emails in the ITD category, 92 in the OAA category and 69 in the OSL category. There are about 16 to 21 emails for every sub-category. As shown in the Table 1, the emails are written to incorporate practical technical challenges (TC: Technical Challenges) encountered in real-world emails: ABR - Abbreviations, SPL - Spelling Errors, SYN - Synonymy, PLY – Polysemy.

In Table 1, column ABR represents the number of emails containing abbreviations in a particular sub-category. Similarly, columns SPL, SYN and PLY represents the number of emails in a sub-category with technical challenges Spelling Errors, Synonym and Polysemy respectively. Overall there exists an abbreviation in 114 email messages, spelling error in 42 email messages, synonym in 28 email messages and polysemy in 10 email messages. Column 0 TC shows the number of emails containing none of the 4 technical challenges. Similarly, column 1 TC, 2 TC and 3 TC shows the number of emails containing any 1, 2 or 3 of the 4 technical challenges respectively. For example, WFO sub-category in ITD class contains a total of 20 email messages. Out of these 20 email messages, 10 emails contain no technical challenge (0 TC). The rest 10 emails contains 1 technical challenge each (1 email contain abbreviations, 3 emails contain spelling errors, 3 emails contain synonym and 1 email contains polysemy). Overall there exists 101 email messages with 0 technical challenge, 117 email messages with any 1 technical challenge, 35 email messages with any 2 technical challenges and 3 email messages with any 3 technical challenges.

2.2 Solution Approach and Research Framework

Figure 1 shows the proposed solution approach and research framework. The overall architecture consists of several building blocks and multiple steps which are explained in the below sub-sections.

2.2.1 Text Pre-Processing

We use the NLTK[2] library for most of our text pre-processing. NLTK has a rich set of Python programs and functions for processing natural language and human language data. We create a text processing pipeline starting from tokenization. We first tokenize the all the emails in our corpus using nltk.tokenize.word_tokenize() method and convert every token to lowercase. We apply lowercase conversion as we do not make use of any linguistic feature which makes use of capitalization information. Numbers and punctuation are also removed as we do not use any features based on numbers and punctuations. All white spaces (tabs, newlines and extra spaces) are trimmed to a single space character. Then we remove every token in the email that is present in the stop-words corpus of the NLTK library (these are standard and general stop words such as and, or, the). However, we also create a domain specific stop word list based on our application requirements. We then utilize the WordNetLemmatizer() method which uses the built-in Morphy method to lemmatize if the word can be found in the WordNet database. We do not apply word stemming as we notice in our target application domain that the context of a sentence

[2] http://www.nltk.org/

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

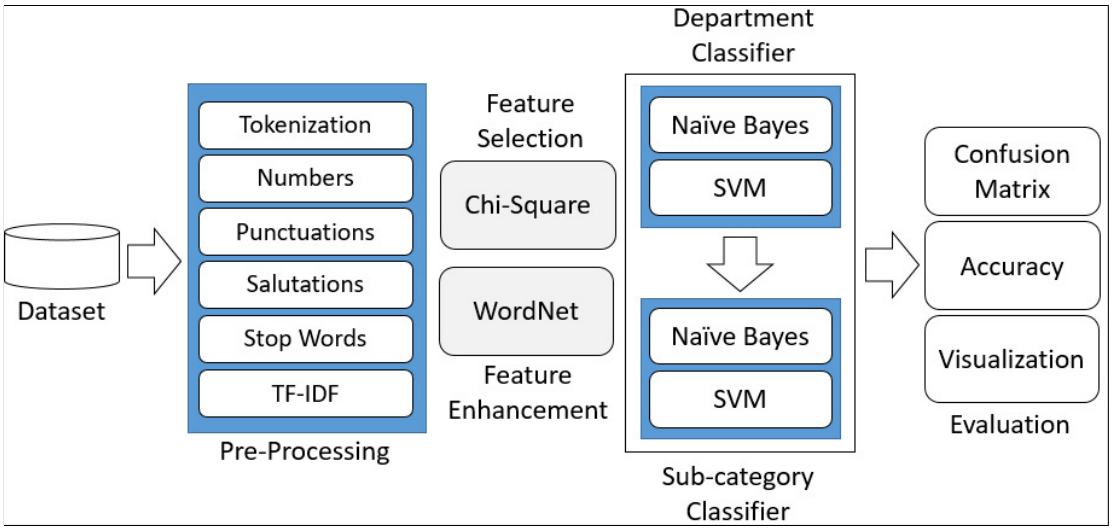

Figure 1. Architecture diagram for the solution approach and research framework

Rank	Feature	χ^2 Values	Rank	Feature	χ^2 Values	Rank	Feature	χ^2 Values
1	add	37.64	11	event	9.83	21	provide	5.68
2	book	22.88	12	help	9.77	22	quiz	5.54
3	cancel	16.45	13	id	9.77	23	registration	5.42
4	cancellation	15.88	14	list	9.61	24	reimbursement	5.39
5	card	13.46	15	login	9.15	25	room	5.25
6	clicker	12.58	16	major	7.83	26	semester	5.17
7	connect	11.19	17	meeting	7.08	27	take	5.15
8	course	10.59	18	offer	7.00	28	timing	5.09
9	detail	10.26	19	password	5.96	29	wifi	5.08
10	drop	9.94	20	projector	5.71	30	work	4.89

Table 2. Chi Square values of Discriminatory Features

is often lost which could negatively impact the precision and recall. In our application domain which consists of students and faculty members (primarily students) sending emails to support functions within a university, there are several salutations like: sir, mam, greetings, hi, dear, hey, hello, good morning, good afternoon, good evening, respected. We remove such salutations as they are not discriminatory in our domain. We also remove signatures like: best regards, thanks, regards, warm regards, kind regards, regards , cheers, many thanks, thanks and regards, sincerely, ciao, best, thank you, talk soon, cordially, yours truly, thanking you, yours thankfully, yours sincerely, thankfully, best wishes. We compute the tf-idf scores (term frequency, inverse document frequency) for every unique term in the corpus. For the tf-idf computation, we use the scikit-learn[3] library which is a machine learning library in Python.

2.2.2 Solutions to Overcome Technical Challenges

Spelling Correction: In our dataset, we mainly checked two different techniques for performing spelling corrections. The first technique locates a correction c, from all the possible candidate corrections. The correction c is selected in such a manner that given the original word w, the following probability value is maximum:

$$argmax_{c \epsilon candidates} P(c|w) = argmax_{c \epsilon candidates} P(c)P(w|c)/P(w)$$

A large English text word corpus is formed from the excerpts of book obtained from Project Gutenerg[4]. Project Gutenberg is repository of 56,000 free eBooks. We selected books of various

[3] http://scikit-learn.org/stable/
[4] https://www.gutenberg.org/

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

Rank	Feature	χ^2 Values	Rank	Feature	χ^2 Values	Rank	Feature	χ^2 Values
1	battery	17.56	6	id card	10.17	11	password	5.75
2	card	16.89	7	login	8.57	12	projector	5.38
3	clicker	15.26	8	login detail	7.71	13	remote	5.11
4	detail	14.34	9	lose	7.40	14	wifi	4.81
5	id	13.13	10	lose id	5.84	15	wifi login	4.53

Table 3. Chi Square values of ITD Discriminatory Bigram Features

disciplines from project Gutenberg. Additionally, we used the list of common English words provided by Wiktionary[5]. Wiktionary is a multilingual free dictionary of all words in all languages. We first calculated the prior probabilities, P(c) of each word c from the corpus. We removed P(w) from formula as the value of P(w) would come out to be the same for every other candidate. We computed P (w—c) by calculating the edit distance of w and c. This method does not take into consideration the context of misspelled word. For Example: "I want an appl" gets corrected to " I want an apply". However, for our objective, context sensitivity is also important. Hence, we used Google's 'Did you mean?' feature. We queried all misspelled email messages from our dataset and downloaded the corresponding suggestion page from Google's 'Did you mean?' feature and scraped it.

Polysemy: Polysemy refers to the simultaneous occurrence of multiple meanings for a single term. In many cases, the meanings belong to completely different contexts. For example: Term 'apple' can refer to the apple fruit or Apple the company. Therefore, it is important to handle polysemy so that the term always get correct weightage as an incorrect weightage might lead to a misclassification. Hence, in order to tackle the problem of polysemy, there is a need to learn the context of each sentence. To handle polysemy, one possible solution was to try accounting for words that enclose the specified word. For example: 'reading book' and 'book room' both contain the term 'book' but the context is different. Hence, to consider the context, we also included the words enclosing the polysemy term. For example, in 'reading book' and 'book room', we took into account the words 'reading' and 'room' as well so that we can perceive that there are two different phrases which are completely different in the contextual space. To implement this phenomenon of considering enclosing words, we considered bi-grams along with the singular terms. The inclusion of bi-grams help us widen the scope of how to visualize each email message. Now, we can derive more information by looking at the adjacent words to a given term.

Therefore, if we now receive an email message regarding reading a book and another email message regarding booking of a room, we will increment the count vector of term 'book' twice. However, the count of phrase 'book room' will increment once and count of phrase ''reading book' will increment once. This would facilitate in improved classification as a new email message about reading a book will not be misclassified because the probability of such email message containing reading and book terms adjacent to each other would be higher than the probability of containing phrase book and room terms next to each other. Thus, this technique of considering bi-grams into consideration solves the problem of polysemy. Also, higher word phrases such as tri-grams, 4-grams or 5-grams would further increase the accuracy of the classification process. However, in this work, we have considered only singular terms and bi-grams.

Synonym: Synonymy - Synonymy is a classic natural language processing issue that occurs in the domain of text classification. To address the synonym issue, we compute a word similarity metric. We use the similarity metric based on Wu Palmer similarity. The WordNet[6] library was used from the NLTK corpus. WordNet is a lexical database for the English language (Miller, 1995). It groups English words into sets of synonyms called synsets (Miller, 1995). These synsets are used to find closely related words of every word in the new incoming email. Now each of the synsets for every word is compared with each feature in the dataset using the Wu Palmer similarity which is present as a functionality in the wordnet library. A threshold of similarity value is pre-decided

[5] https://www.wiktionary.org/
[6] https://wordnet.princeton.edu/

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

by us. In our case, 0.85 was the pre-decided value which signifies a very high similarity metric. Path similarity computes shortest number of edges from one word sense to another word sense, assuming a hierarchical structure like WordNet (essentially a graph). In general, word senses which have a longer path distance are less similar than those with a very short path distance. Therefore words like internet and wifi or refund and reimbursement will have a very high similarity value which allows us to account for these highly similar words in the classification task. For example, in our dataset similarity(refund, reimburse) wup_similarity is 0.88 and similarity(Wifi, internet) wup_similarity is 0.857.

Abbreviations: Email messages often contains abbreviations as this leads to increase in speed of message exchange. Since the message receiver is also aware of common abbreviations, this model works. On the contrary, computer would treat the full and abbreviated form of text as two different terms. In order to overcome this problem, we need a list of abbreviations. This work deals with the email messages within a university context only. Universities generally have a well defined set of limited lexicons which are used widely. For example, departments codes, course codes etc. Hence, we manually created a dictionary with mappings of the most popular abbreviated terms and their expansions.

The manually created dictionary is further used in classifications in two ways:

1. Before lemmatizing a term, we look up the term in abbreviation list and if the term is present, we exempt the term from lemmatization process as the term is already present in correct format and there is no need for lemmatization.

2. When we create count vectors, and encounters an abbreviated form, we first map the abbreviated form with the expanded form. Next, we update the counter vectors for abbreviated term as well as for all the terms in the expanded form. For Eg: if the word OSL is present in an email, then we first map it with its expanded form i.e. Office of Student Life. Next, the counter for abbreviated form OSL will be incremented. Also, the words Office, Student, Life would get accounted for in the bag of words model and counts of each word (office, student and Life) will get incremented. This ensures that no matter whether we receive an abbreviation/expanded form in message, they will get accounted for in the classification process.

2.2.3 Feature Selection and Enhancement

Feature selection is the mechanism of selecting a subset of relevant features which are supplied as input in the machine learning model. It is one of the most important pre-processing steps in machine learning frameworks. There can be multiple features in data, some of them can be relevant but some others can be redundant features or irrelevant features. Feature selection techniques tries to remove such redundant and irrelevant features and select the features which are most discriminatory. This helps in selection of informative features which results in better prediction accuracy values. There exists a wide range of feature selection techniques. In our experiments, we used Chi-Square feature selection technique.

Chi-Square Chi-Square is a statistical test to determine the dependency of two variables (Yang and Pedersen, 1997). In machine learning models, there are various features and a target class. Chi-square test is used to measure the existence of relationship among various features with target class. The features with higher relationship acts as discriminatory features.

2.2.4 Classifiers

There exists a wide range of machine learning classification algorithms. In our experiments, we used Naive Bayes and SVM learning algorithms for classification:

Naive Bayes: Naive Bayes classifier is a supervised machine learning algorithm (McCallum et al., 1998). It belongs to the family of simple probabilistic classifiers which are based on Bayes theorem. Naive Bayes classifier is one of the most widely used text classification algorithms. It works on the principle of word counts and is highly scalable.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

SVM: Support vector machines are supervised learning models which can be used for classification or regression problems. SVM works on the principle of creating hyperplanes. Each data point is considered as a p-dimensional vector and the goal is to find whether points can be separated with a (p-1)-dimensional hyperplane. SVM has been used in various kinds of text classification problems and has been proved to be among the best classification algorithms(Smola and Schölkopf, 2004).

We used Naive Bayes and SVM classifiers for two types of classification: (1) Department classification (ITD, OSL and OAA), and (2) Sub-category level classification (14 sub-categories).

2.2.5 Evaluation

Model evaluation is one of the most important step in machine learning pipeline. In this work, we used confusion matrices and accuracy measure for model evaluation. A confusion matrix is a precise, tabulated form of representing prediction results obtained in a machine learning classification task. It represents the number of correctly and incorrectly classified instances by a machine learning algorithm. The rows of the confusion matrix lists all the predicted classes and the columns of the confusion matrix lists all the actual classes. The diagonal elements in a confusion matrix represent number of correctly classified instances, i.e. the instances were predicted to the actual class only by the learning algorithm. The elements other than diagonal elements in the confusion matrix represents the number of incorrectly classified instances. We represent confusion matrices of both Naive Bayes and SVM classifiers for department level and sub-category level classification. Another evaluation parameter used in this work is Accuracy. Accuracy is a metric to judge the goodness of machine learning classification model. It is the ratio of correctly classified instances to the total number of instances in test set. We calculate accuracy values for department level and sub-category level classifications. In addition to confusion matrices and accuracy tables, we also used visualizations to represent our dataset and results. We used various box-plots to show the overall spread of values in discriminatory features. This representation of spread enables the better understanding of our dataset and feature distributions.

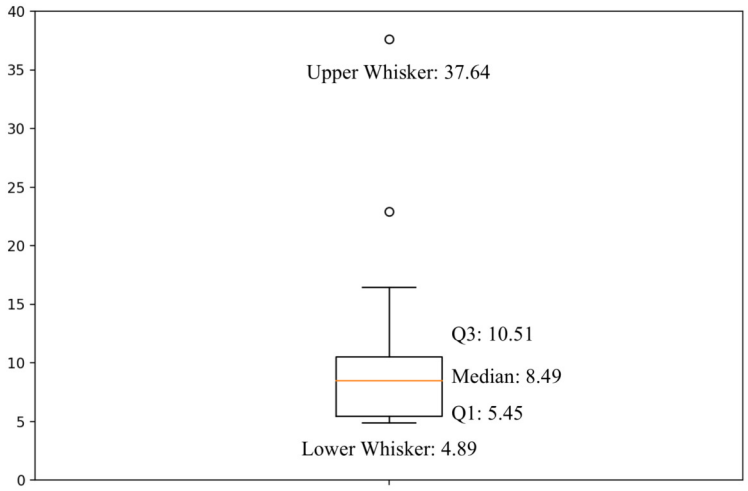

Figure 2. Boxplot of Chi-Square Values of 30 Discriminatory Features

3 RESULTS

3.1 Discriminatory Features

Every term after pre-processing (general stop word removals, domain specific stop word removal, and lemmatization) is a feature in our text classification problem. The discriminatory power of a feature is the relative usefulness or relevance of the feature for the classification task. We use the chi-square score

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

Category	Sub-Category	Term TF-IDF-Score			
ITD	WFO	internet 0.55	wifi 0.56	connect 0.37	laptop 0.41
	LOD	reset 0.53	portal 0.64	password 0.51	detail 0.64
	CLK	presentation 0.34	require 0.57	clicker 0.48	remote 0.62
	CLE	mic 0.53	issue 0.3	projector 0.52	speaker 0.57
	IDC	lose 0.39	buy 0.53	find 0.4	id 0.55
OAA	ADC	course 0.31	add 0.56	join 0.49	permission 0.45
	DPC	drop 0.69	remove 0.69	course 0.32	needful 0.46
	CTM	slot 0.47	clash 0.49	timetable 0.43	timing 0.62
	COF	list 0.43	major 0.59	next 0.42	semester 0.41
	CDT	registration 0.59	date 0.59	timing 0.49	open 0.33
OSL	RBK	event 0.47	book 0.44	lecture 0.47	onwards 0.32
	BCL	cancel 0.42	inconvenience 0.47	book 0.43	
	MSD	meeting 0.52	book 0.16	fest 0.46	meet 0.25
	RMB	reimbursement 0.48	travel 0.44	approve 0.3	receive 0.43

Table 4. TF-IDF scores of few terms in the dataset

(based on the chi-square statistical test) as the metric to compute the feature importance or discriminatory power of a feature. Tables 2 and 3 shows the chi-square scores of various terms for both the levels of the classifier (department level and sub-category level). Table 2 reveals that there are several terms in the dataset which provides a strong signal for determining the results of the department level classifier. Table 2 can be viewed as a relative comparison of the discriminatory power of the top 30 features while predicting the department of the incoming email. A lower value of the chi-square score shows lack of dependence between the feature and the class and a higher value shows correlation. Few features with the highest discriminatory power for the first classifier are: add, book, cancel, cancellation, card, clicker, connect, course, detail, drop, event, help, id, list, login and major. For example, there is a string relation between the term login and ITD. Similarly, there is a strong correlation between the term major and OAA. The chi-square score value of add and book is the highest and is above 20. Terms like timing, wifi and work have a discriminatory power but is low. We observe from Table 2, that terms like password and projector have a chi-square score of 5.96 and 5.71 respectively. Terms like password and projector are indicators of the ITD class. The term registration has a chi-square score of 5.42 and is an indicator of the OAA class. Table 3 presents results for the second level classifier and lists the relative chi-square scores of the terms which are indicators of the ITD classifier. Table 3 reveals that terms like battery, card clicker, detail, id, id card, login, login detail and lose have high discriminatory power of the ITD class. Table 3 shows both the unigrams as well as the bigrams. We observe that the range of chi-square score values varies from a minimum of 4.53 to a maximum of 17.56.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

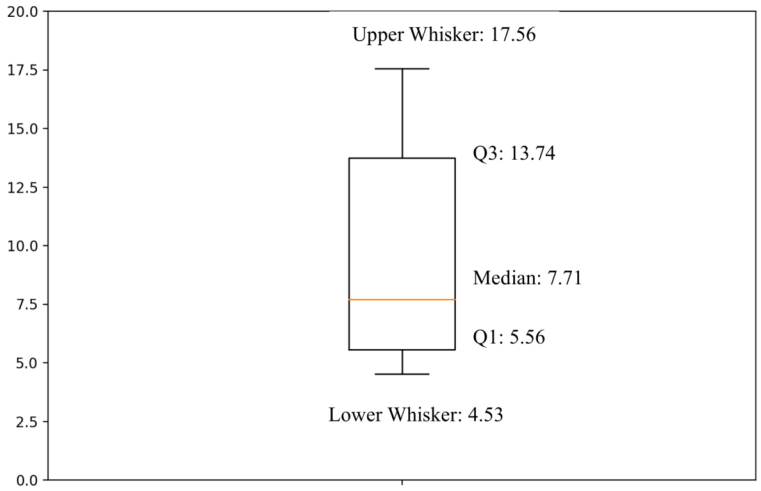

Figure 3. Boxplot of Chi-Square Values of 15 Discriminatory Features in IT Department

3.2 TF-IDF Scores

TF-IDF (term frequency–inverse document frequency) is a weighting factor used in information retrieval for computing the relative importance of a term in the document within a document collection or corpus Aizawa (2003). The TF-IDF score is proportional to the frequency of the term in the document and also takes into account how common the term is in the document collection or corpus and not just the given document Aizawa (2003). Table 4 displays the TF-IDF values of several terms (taken a sample from all the unique terms in the corpus) in the document collection of our experimental dataset. Table 4 reveals that the tf-idf value of the term drop is 0.69. This is because the term drop is occurring several times within a small number of documents (belonging to a particular like : OAA) and hence results in a high discriminatory power for the documents in which it occurs. We observe that the term meet has a relatively low tf-idf score of 0.25 which means that the term is occurring in a relatively large number of documents and also occurring fewer times for the given document resulting in a less pronounced relevance or identification signal for a class.

Similarly, terms such as internet and wifi have high tf-idf scores in WFO sub-category whereas detail, portal have higher relevance in LOD sub-category. Also, terms such as reset, password, portal, detail and remote have high tf-idf scores in sub-categories of ITD class. This represents that these terms are highly relevant for predicting ITD class. On the other hand, terms such as issue, connect and presentation have low tf-idf scores in ITD category representing low relevance while predicting ITD class. For OAA class, terms such as add, remove, drop, major, date, registration have high tf-idf score representing higher relatedness for predicting OAA class. We observe that term timing occurs in both CTM and CDT sub-category. However, its tf-idf score for sub-category CTM is higher than its tf-idf score for sub-category CDT. This represents that relevance of term timing is high in both CTM and CDT but the relevance is more to CTM than CDT sub-category. For OSL class, terms such as meeting and fest have high tf-idf score for MSD sub-category whereas lecture and event terms have high tf-idf score for RBK sub-category.

3.3 Chi Square Test Box Plot Visualization

Figures 2 and 3 shows the chi-square values for the top 30 discriminatory features for the department level classification and the chi-square values for the top 15 discriminatory features for the ITD sub-category level classification. Forman et al. conduct an empirical study on various feature selection metrics for text classification (Forman, 2003). Forman et al. mention that chi squared is a commonly known metrics and a statistical test which can be used for feature selection (Forman, 2003). We compute the chi-square values for all the features (unigrams and bigrams) in our dataset for both the department level classification

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

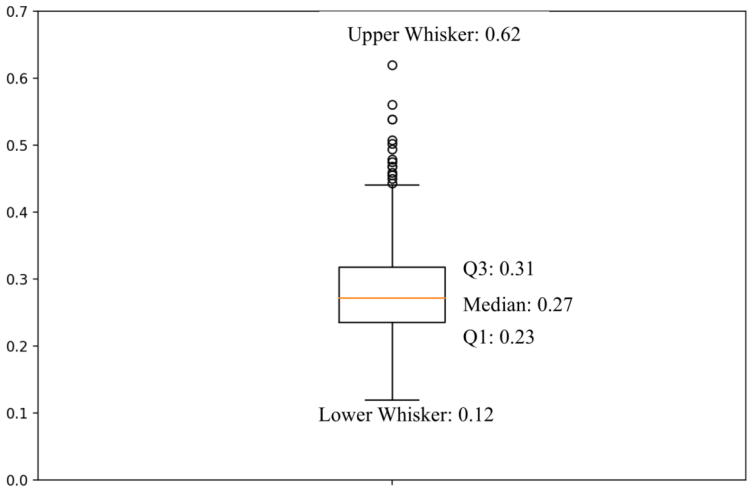

Figure 4. Boxplot of IDF of all terms.

and sub-category within a department level classification. We compute the chi-square between each feature and the class. The score is then used to select the top 30 features (top 30 highest values). Since the chi-square test measures the dependence between stochastic variables, we use the chi-square test based feature selection to identify relevant as well as irrelevant features for our classification problem. Figures 2 and 3 displays the chi-square score of the top features through their quartiles. Figures 2 and 3 are useful for understanding the variation in the chi-square score for the most relevant features. The box plots in Figures 2 and 3 shows the dispersion or spread in the chi-square values. The median value for the chi-square score for the top 30 discriminatory features at the department level classification is 8.49. Figures 2 reveals that the minimum value for the score is 4.89 and the maximum value is 37.64 which clearly shows variation in the discriminatory power of the features. The box plot in Figure 3 is useful from the perspective of understanding the distributional characteristics of the chi-square scores and shows that there is a wide range of scores. Both the box plots in Figures 2 and 3 that there are several values in the upper and lower whiskers representing scores outside the middle 50%. We observe that the shape and positions of various points in both the box plots are different in-terms of the median values, range and the distribution. The median for box plot in Figure 2 is at a relatively higher level than the median for the box plot in Figure 3. Also, we observe that the four sections in the box plots are uneven in size and hence the changes in the chi-square values (representing the relevance of a feature) are variable.

Table 5. Department Confusion Matrix-NB

	Naive Bayes		
	ITD	OAA	OSL
ITD	90	0	0
OAA	2	82	0
OSL	2	2	62

3.4 Confusion Matrix

Tables 5 and 6 displays the confusion matrix to describe the performance of the Naïve Bayes and SVM classification model for the department level classification. We create the confusion matrix as our dataset is annotated and we know the true values of every instance. There are three actual and predicted classes for the department level classification task: ITD, OAA and OSL. The row of the confusion matrix represents the actual class and the column represents the predicted class. Table 5 reveals that there were 90 instances

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

Table 6. Department Confusion Matrix-SVM

	SVM		
	ITD	OAA	OSL
ITD	90	0	0
OAA	2	82	0
OSL	0	0	66

Table 7. ITD Confusion Matrix - NB

	Naive Bayes				
	WFO	CLK	CLE	IDC	LOD
WFO	18	0	0	0	0
CLK	0	18	0	0	0
CLE	1	1	17	0	0
IDC	0	0	1	15	0
LOD	0	0	0	0	19

Table 8. ITD Confusion Matrix - SVM

	SVM				
	WFO	CLK	CLE	IDC	LOD
WFO	18	0	0	0	0
CLK	0	18	0	0	0
CLE	1	1	17	0	0
IDC	0	0	1	15	0
LOD	0	0	0	0	19

Table 9. OAA Confusion Matrix - NB

	Naive Bayes				
	DPC	ADC	COF	CDT	CTM
DPC	15	0	1	0	0
ADC	0	12	3	1	0
COF	0	0	18	0	1
CDT	0	0	2	11	3
CTM	0	0	0	1	16

Table 10. OAA Confusion Matrix - SVM

	SVM				
	DPC	ADC	COF	CDT	CTM
DPC	15	1	0	0	0
ADC	0	16	0	0	0
COF	0	0	19	0	1
CDT	0	0	0	15	0
CTM	0	0	0	1	16

of ITD and all were correctly classified. True positives are cases which are correctly classified. For example, all emails which were ITD and were predicted as ITD will be true positives. True negatives are cases which were not ITD and were not classified as ITD. Accuracy computed by summing the value of true positives and true negatives and dividing it by the total number of instances in the dataset. We present the results in the form of confusion matrix as our problem is a multi-class classification problem and not just a binary class problem and also we our objective was to study both correct classification and

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

Table 11. OSL Confusion Matrix - NB (NaiveBayes)

	Naive Bayes			
	RBK	RMB	MSD	BCL
RBK	18	0	0	0
RMB	0	16	0	0
MSD	0	0	17	0
BCL	1	0	0	14

Table 12. OSL Confusion Matrix - SVM (Support Vector Machines)

	SVM			
	RBK	RMB	MSD	BCL
RBK	18	0	0	0
RMB	0	16	0	0
MSD	0	0	17	0
BCL	0	0	0	15

misclassification with respect to every class. Tables 5 and 6 reveals the number of cases where the classifier is going wrong. For example, there are two instances of OAA which were wrongly classified at ITD by the Naïve Bayes classifier. Similarly, there are 2 instances of OSL which are wrongly classified as ITD and 2 instances of OSL which are misclassified as OAA. Tables 6 reveals the number of misclassifications. Tables 6 shows that there are 2 instances of OAA which are misclassified by the SVM classifier to ITD. Recall for a particular class is a measure of the probability of the correctly classified instances with respect to all the examples belonging to the particular class. From Tables 5 and 6, we can infer a high recall values for both all the three classes in the dataset. We also observe a high precision as a high precision represents cases which are labelled as positive with respect to a class and are indeed positive with respect to the class. Table 6 reveals that all the 66 instances of OSL are correctly classified as OSL by the SVM classifier.

Table 7, 8, 9, 10, 11 and 12 displays the confusion matrices to describe the performance of Naïve Bayes and SVM classifier for sub-category level classification. Table 7 and 8 shows the confusion matrices for Naïve Bayes and SVM classifier for 5 sub-categories (WFO, CLK, CLE, IDC and LOD) of ITD class. Table 7 and 8 reveals that for sub-categories WFO, CLK and LOD, both Naïve Bayes and SVM machine learning algorithms classify all the instances correctly whereas 2 instances of CLE sub-category are misclassified (one as WFO and another as CLK) and 1 instance of IDC sub-category is misclassified as CLE. Similarly, Table 9 and 10 shows the confusion matrices for Naïve Bayes and SVM classifier for 5 sub-categories (DPC, ADC, COF, CDT and CTM) of OAA class. Table 9 depicts that out of 92 total instances in ITD class, 72 instances get correctly classified across its sub-categories by Naïve Bayes classifier. On the other hand, Table 10 shows that SVM classifier correctly classifies 81 instances across 5 subcategories. Table 11 and 12 presents the confusion matrices for Naïve Bayes and SVM classifier for 4 sub-categories (RBK, RMB, MSD and BCL) of OSL class. Table 11 shows that out of 66 total instances in OSL class, 65 instances get correctly classified by Naïve Bayes classifier whereas SVM classifier correctly classifies all 66 instances across 4 sub-categories.

Table 13. Accuracy Table - Department Level

	ITD	OAA	OSL	Overall
Naive Bayes	1	0.976	0.939	0.975
SVM	1	0.976	1	0.991

3.5 Classification Accuracy

Tables 13, 14, 15 and 16 shows the accuracy results for the department level (ITD, OAA or OSL) and the sub-category level (categories or topics within a particular department). Tables 13, 14, 15 and 16 presents the accuracy results for both the classifiers: NaiveBayes and SVM. Table 13 reveals that the overall

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

Table 14. Accuracy Table - ITD

	WFO	CLK	CLE	IDC	LOD	Overall
Naive Bayes	1	1	0.894	0.937	1	0.966
SVM	1	1	0.894	0.937	1	0.966

Table 15. Accuracy Table - OAA

	DPC	ADC	COF	CDT	CTM	Overall
Naive Bayes	0.937	0.75	0.947	0.687	0.941	0.857
SVM	0.937	1	0.95	1	0.941	0.964

Table 16. Accuracy Table - OSL

	RBK	RMB	MSD	BCL	Overall
Naive Bayes	1	1	1	0.933	0.984
SVM	1	1	1	1	1

accuracy for the NaiveBayes algorithm at the department level classification task is 0.975. The overall accuracy for the SVM learning algorithm at the department level classification task is 0.991. We perform 4 fold cross validation in all our experiments to compute the overall accuracy. In 4 fold cross validation we randomly partition the dataset into 4 equal sized sub samples. After partitioning the data, one of the partition is used as the testing data and the remaining 3 samples are used as the training dataset. We use cross validation technique to evaluate our classifier as it minimizes biases in the training and test dataset. We observe that SVM outperforms NaiveBayes by a small margin. SVM results in 100% accuracy for the ITD and OSL class. NaiveBayes results in best performance for the ITD class in comparison to OAA and OSL.

Table 14 presents the accuracy results of both the machine learning classifiers: NaiveBayes and SVM across 5 sub-categories (WFO, CLK, CLE, IDC and LOD) of ITD class. We observe that both the machine learning classifiers NaiveBayes and SVM achieve similar accuracy results across all 5 sub-categories.The overall accuracy for ITD class is 0.966 for both learning algorithms. The table reveals that both the classifiers results in 100% accuracy for WFO, CLK and LOD sub-categories, 89.4% accuracy for CLE sub-category and 93.7% for IDC sub-category. Table 15 shows the accuracy of NaiveBayes and SVM classifiers across 5 sub-categories (DPC, ADC, COF, CDT and CTM) of OAA class. Table 15 reveals that overall accuracy for NaiveBayes classifier in OAA class is 85.7% whereas SVM outperforms NaiveBayes learning algorithm and results in 96.4% overall accuracy. Among the 5 sub-categories in OAA, the best performance of 100% is achieved by SVM classifier for ADC and CDT sub-categories. For sub-category COF, SVM classifier performs slightly better and results in 0.95 accuracy whereas NaiveBayes classifier achieves 0.947 accuracy. For DPC and CTM sub-categories, both SVM and Naive Bayes classifiers results in same accuracy values of 93.7% and 94.1% respectively.

Table 16 presents the accuracy results of Naive Bayes and SVM learning algorithm across 4 sub-categories (RBK, RMB, MSD and BCL) of OSL class. The table reveals that overall accuracy in OSL class for NaiveBayes learning algorithm is 0.984 whereas SVM classifier results in 100% accuracy. For RBK, RMB and MSD sub-categories both learning algorithm are able to achieve accuracy result of 100%. For BCL sub-category, Naive Bayes algorithm results in 0.933 accuracy whereas SVM outperforms Naive Bayes algorithm and achieves 100% accuracy. For OSL class, SVM machine learning algorithm achieves 100% accuracy for all the 4 sub-categories resulting in 100% overall accuracy.

4 DISCUSSION

4.1 Web-Based Application

In addition to conducting machine learning experiments, we also developed a web application after taking inputs from the users. A web application or a mobile application are the two possible approaches to deploy an automatic email response suggestion tool within an enterprise and make practical use of it. We

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

developed a web application using Flask[7] which is a micro-framework for Python. We developed our application using Flask as it contains several modules and libraries enabling us to write an application with a focus on our application specific requirements and not concerning ourselves with low-level details like thread management and protocols. We use Gmail API[8] which is a RESTful API and can be used to access Gmail boxes and send emails through it. We use Gmail API as we use Google Apps within our university. The email system used by various support functions within our university is Gmail. The emails are fetched from the Gmail API and the Flask front-end and the system takes the email body and subject to start the process of suggesting replies. Once the ITD, OAA or OSL person opens this interface, they see the next screen, which contains the editing reply option and selecting which reply to send option. Figure 5 shows the snapshot of the front-end of the web application developed by us. As shown in Figure 5, the user sees the various suggestions from the back-end machine learning system and can select the best option and also make text edits in the subject or message body.

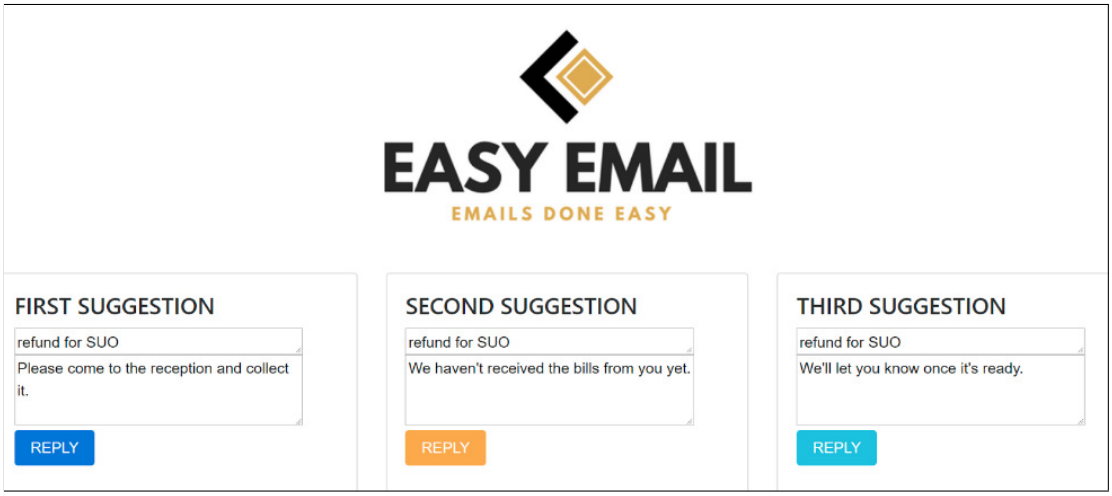

Figure 5. Snapshot of the web application for automatic email response suggestion system

4.2 Threats to Validity

The work presented in this paper is an empirical study consisting of an empirical evaluation and empirically investigated hypothesis and claims. In this section, we discuss how we maximized internal and external validity and present our analysis of the various threats to validity in our experiments. While we try to mitigate various types of threats to validity issues, as mentioned by Siegmund et al., there is an inherent trade-off between internal and external validity Siegmund et al. (2015). One threat to validity is the researcher bias (who does the work) Shepperd et al. (2014), the predictive performance of machine learning classifiers can be influenced by several parameters such as the choice of classifiers by the researchers, dataset used by the researchers as well as reporting protocols citeshepperd2014researcher. One threat to validity is that are the changes in the independent variables (or features) are indeed responsible for the observed variation in the target or dependent variable (email response or suggestion category in our case). In order to mitigate this threat to validity, we created variations in the input dataset and conducted correlation tests between the dependent and independent variable. We extract features from the textual email content and do not perform any link or graph analysis which can be extraneous variables or confounding variables that can also influence the dependent variable (this is one possible threat to validity). To mitigate external validity on whether our results are applicable to other classes or sub-categories, we created 3 categories (ITD, OAA, OSL) and 10 sub-categories. However, more experiments are required to investigate if the study results and approach is applicable to other categories and sub-categories. The dataset was annotated and verified by more than one person (authors this paper) to ensure that the dataset annotation is of high quality and there are no annotation and measurement errors. We also executed the experiments more than once to ensure that there are no errors while conducting the experiments

[7] http://flask.pocoo.org/
[8] https://developers.google.com/gmail/api/

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

and that our results are replicable. While our results shows relationship between the dependent variable (department or sub-category within a department) and independent variables (terms within an email), we believe more experiments on a large dataset and dataset belonging to more categories is needed to strengthen our conclusions that the variables accurately model our hypothesis.

5 CONCLUSION

We present a solution approach for automatically suggesting email responses to short and frequent messages sent to support department and functions within a university. The proposed solution is aimed at building web-based or mobile systems and applications for providing a one tap or click solution for responding to large number of frequent queries from users. We create the first dataset for the novel application of email response suggestion in a university domain and conduct a series of experiments to evaluate the proposed approach. Our approach is a multi-step process consisting of text processing (such as tokenization, stop term removal and lemmatization), feature selection step (suing chi-square test statistics and score), two-level classification (one for the department and one for the sub-category) and performance evaluation. Our experimental results demonstrate the effectiveness of the proposed solution approach. We observe that terms in the email documents can be used as discriminatory features to identify the class of a document (in our case the department and the type of query). Our experimental results reveal discriminatory and non-discriminatory features and shows that the relevance of the terms with respect to their discriminatory power varies across terms. Our experimental results reveal that the chi-square test approach is an effective feature selection technique for the specific problem addressed by us. We observe several technical challenges in the dataset such as abbreviations, spelling mistakes, synonyms and polysemy and propose an approach to provide solutions to the technical challenges. We experiment with two different learning algorithms and observe that SVM outperforms Naïve Bayes. We achieve a classification accuracy of above 85% for all the classes and sub-classes.

REFERENCES

Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, 39(1):45–65.

Al-Alwani, A. (2015). Improving email response in an email management system using natural language processing based probabilistic methods. *Journal of Computer Science*, 11(1):109.

Ayodele, T., Zhou, S., and Khusainov, R. (2009). Email reply prediction: a machine learning approach. In *Symposium on Human Interface*, pages 114–123. Springer.

Di Castro, D., Karnin, Z., Lewin-Eytan, L., and Maarek, Y. (2016). You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 307–316. ACM.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of machine learning research*, 3(Mar):1289–1305.

Graus, D., Van Dijk, D., Tsagkias, M., Weerkamp, W., and De Rijke, M. (2014). Recipient recommendation in enterprises using communication graphs and email content. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1079–1082. ACM.

Henderson, M., Al-Rfou, R., Strope, B., Sung, Y.-h., Lukacs, L., Guo, R., Kumar, S., Miklos, B., and Kurzweil, R. (2017). Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., Corrado, G., Lukács, L., Ganea, M., Young, P., et al. (2016). Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964. ACM.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Shepperd, M., Bowes, D., and Hall, T. (2014). Researcher bias: The use of machine learning in software defect prediction. *IEEE Transactions on Software Engineering*, 40(6):603–616.

Siegmund, J., Siegmund, N., and Apel, S. (2015). Views on internal and external validity in empirical

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018

software engineering. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*, volume 1, pages 9–19. IEEE.

Singh, A., Mishra, D., Bansal, S., Agarwal, V., Goyal, A., and Sureka, A. (2018). Email dataset for automatic response suggestion within a university.

Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222.

Sneiders, E., Sjöbergh, J., and Alfalahi, A. (2017). Automated email answering by text-pattern matching: Performance and error analysis. *Expert Systems*.

Van Gysel, C., Mitra, B., Venanzi, M., Rosemarin, R., Kukla, G., Grudzien, P., and Cancedda, N. (2017). Reply with: Proactive recommendation of email attachments. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 327–336. ACM.

Yang, L., Dumais, S. T., Bennett, P. N., and Awadallah, A. H. (2017). Characterizing and predicting enterprise email reply behavior. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 235–244. ACM.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Icml*, volume 97, pages 412–420.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26531v1 | CC BY 4.0 Open Access | rec: 17 Feb 2018, publ: 17 Feb 2018