© Universiti Tun Hussein Onn Malaysia Publisher's Office

IJIE

Journal homepage: http://penerbit.uthm.edu.my/ojs/index.php/ijie
ISSN : 2229-838X e-ISSN : 2600-7916

The International Journal of Integrated Engineering

Categorizing Natural Language-Based Customer Satisfaction: An Implementation Method Using Support Vector Machine and Long Short-Term Memory Neural Network

Ralph Sherwin A. Corpuz[1*]

[1]Technological University of the Philippines,
Ayala Blvd., Ermita, Manila, 1000, PHILIPPINES

*Corresponding Author

DOI: https://doi.org/10.30880/ijie.2021.13.04.007
Received 18 October 2020; Accepted 26 November 2020; Available online August 2021

Abstract: Analyzing natural language-based Customer Satisfaction (CS) is a tedious process. This issue is practically true if one is to manually categorize large datasets. Fortunately, the advent of supervised machine learning techniques has paved the way toward the design of efficient categorization systems used for CS. This paper presents the feasibility of designing a text categorization model using two popular and robust algorithms – the Support Vector Machine (SVM) and Long Short-Term Memory (LSTM) Neural Network, in order to automatically categorize complaints, suggestions, feedbacks, and commendations. The study found that, in terms of training accuracy, SVM has best rating of 98.63% while LSTM has best rating of 99.32%. Such results mean that both SVM and LSTM algorithms are at par with each other in terms of training accuracy, but SVM is significantly faster than LSTM by approximately 35.47s. The training performance results of both algorithms are attributed on the limitations of the dataset size, high-dimensionality of both English and Tagalog languages, and applicability of the feature engineering techniques used. Interestingly, based on the results of actual implementation, both algorithms are found to be 100% effective in accurately predicting the correct CS categories. Hence, the extent of preference between the two algorithms boils down on the available dataset and the skill in optimizing these algorithms through feature engineering techniques and in implementing them toward actual text categorization applications.

Keywords: Text Categorization, LSTM, SVM, customer satisfaction, natural language processing, machine learning, deep learning, ISO 9001

1. Introduction

Organizations nowadays are heavily impacted by rapid technological change, proliferation of knowledge-based economies, and globalization. As a result, customers have become more educated yet demanding. In order to thrive with these challenges, organizations are compelled to establish Quality Management Systems (QMS) in order to leverage their strategic performance and to consistently meet the demands of their customers [1]. One of the major requirements of a QMS is customer focus, which requires organizations to promote and ensure customer satisfaction (CS) [2]. CS is defined as the perceived degree of fulfillment of customers' expectations [3]. Shown on Fig. 1 is a conceptual model of CS as elaborated in ISO 10004:2018 [4].

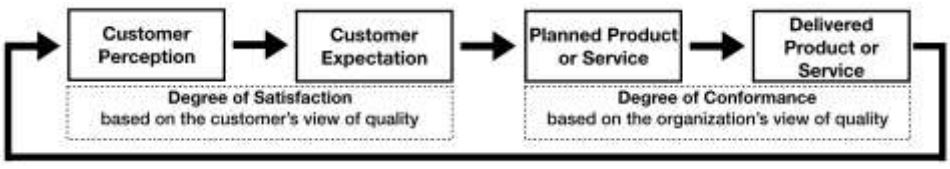


Fig. 1 - CS Model [4]

*Corresponding author: ralpsherwin_corpuz@tup.edu.ph

CS is an equilibrium between customer's expectations and perception of "quality" products or services, and organization's ability to plan and deliver these "quality" products or services. For this purpose, it is important that an organization should determine the degree of CS and at the same time, ensure the effective implementation of a QMS in conformance with international standards, such as the ISO 9001:2015. CS has a significant impact into organization's social branding. High CS equates good reputation, while low CS causes loss of customer trust. To manage these challenges, organizations are required to determine the methods in monitoring and evaluating CS [1], [4]. Likert scaled-surveys are considered to be the most popular CS method, however, there is a recent unprecedented shift in obtaining CS data using a wide-range of internet-based platforms, such as social networking sites, emails, group chats, fora, etc., where customers can freely express their thoughts and feelings written in natural languages. These text-based data are considered significant because they provide insights about customers' emotions, preferences, and the extent of satisfaction or dissatisfaction on organizational performance. Ultimately, these results are used as basis toward the implementation of improvement initiatives, such as introduction of breakthrough processes, products or services that further enhance organizational performance and ensure CS [5], [6].

Unfortunately, text-based CS data are highly dimensional, unstructured and complicated, particularly those with massive datasets. Hence, artificial intelligence (AI) approaches are explored to facilitate automatic extraction and analysis [6] [7]. Text categorization is an area of natural language processing (NLP), a subset of AI, that deals with computational process of labelling texts according to pre-defined features and thematic categories [8]. Each text is represented by a vector and is manipulated through the use of feature engineering techniques. Afterwards, machine learning or deep learning algorithms are utilized in order to fine-tune certain parameters toward the design of text categorization models. The resulting model then automatically finds categories from enormous datasets, wherein such feature essentially makes text categorization a popular NLP application used in a wide-range of domains [9], [10]. Among the most popular algorithms employed for text categorization purposes are the Support Vector Machine (SVM) and Long Short-Term Memory (LSTM) Neural Network.

SVM is a popular categorization algorithm introduced by Vapnik [11]. It is considered popular because of its solid mathematical foundation, which is based in the principle of risk minimization. Likewise, SVM has been proven to have excellent generalization performance, particularly in highly dimensional applications, such as text categorization [5], [12], [13]. Illustrated on Fig. 2 is a representation of a SVM used for binary categorization. In its basic context, SVM determines the best hyperplane that separates the data points between categories. The separating hyperplane is considered to be the best decision boundary for an SVM if it has the largest margin between categories. The data points that are closest to the hyperplane are called *support vectors*, which are found near the margin boundary. *Support vectors* can be further categorized as type 1 or type -1, labeled as "+" or "-", respectively [14], [15].

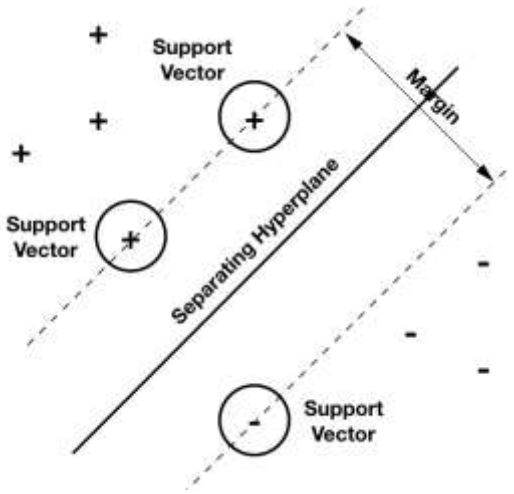


Fig. 2 - SVM Model for Binary Categorization

The SVM decision function, which determines the maximum distance of the margin, is computed using the following formula [16]:

$$f_h\,(x) = sgn\left[\sum_p \alpha_p y_p\, k_f\left(x_p, x\right) + b\right] \tag{1}$$

where "x" is universal test data point; "k_f" is the kernel function, ", "x_p" and "y_p" are x and y data points, respectively; "α" is a Lagrange multiplier; and "b" is the bias. Relatively, SVM is widely-studied in text categorization particularly for customer satisfaction applications, such as those applied to hotel reviews [7], customer relationship management (CRM) [5], [12], website service quality [17], Software-as-a-Service (SaaS) [18], product reviews [19], short text messages [20], among others. Results on these studies generally revealed that SVM has remarkable performance in cases

where datasets have noisy, imbalance or mislabeled categories [7], [20]. SVM has also been found to have better categorization accuracy than recurrent neural network (RNN) [7], random forest, logistic regression [12], artificial neural network (ANN), k-nearest neighbors [17], [20], and other traditional algorithms.

Meanwhile, LSTM is arguably the most popular type of RNN that is capable of learning long-term dependencies between time steps of sequence data. What makes LSTM special is that it can address dissipating gradients, which is a common problem among neural networks, including RNN. Depicted on Fig. 3 is a sample *memory cell* that represents a unit of an LSTM neural network. The cell remembers the information "c_{t-1}, o_{t-1}" and use these as its current, *memory cell state* in random time step "t". Afterwards, it determines the next sequence by updating its *memory cell state* "c_t" and computing the output "o_t" also known as the *output state*. The status of the cell is updated through the use of *gates* that regulate the flow of information into and out of the *memory cell*. It has input gate "i_g" that controls the updates; forget gate "f_g" that forgets the information; *cell candidate* "c_c" that adds information; and output gate "o_g" that controls the output. The arrows in the cell indicate the directional flow of information [21]:

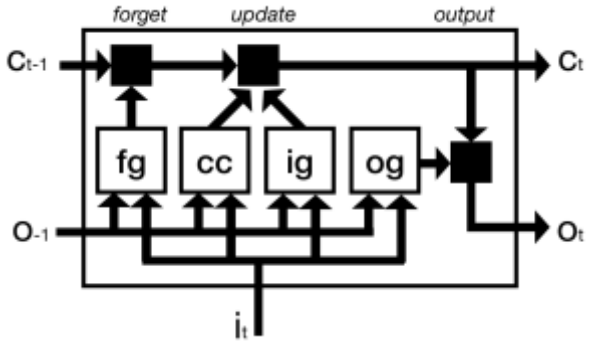


Fig. 3 - LSTM memory cell and gates

The inputs "i_t" are composed of three types of concatenated learning weights called input weights "i_w", recurrent weights "r_w" and bias "b", which have the following matrices [21]:

$$I_w = \begin{pmatrix} I_{w\,ig} \\ I_{w\,fg} \\ I_{w\,cc} \\ I_{w\,og} \end{pmatrix}, R_w = \begin{pmatrix} R_{w\,ig} \\ R_{w\,fg} \\ R_{w\,cc} \\ R_{w\,og} \end{pmatrix}, b \begin{pmatrix} b_{ig} \\ b_{fg} \\ b_{cc} \\ b_{og} \end{pmatrix}, \tag{2}$$

The *memory cell state* at arbitrary time step "t" is expressed in the following formula:

$$c_t = fg_t \odot c_{t-1} + ig_t \odot cc_t \tag{3}$$

where "⊙" is the Hadamard product or result of elementwise multiplication of vectors. Meanwhile, the *output state* at time step "t" is determined by the following formula:

$$o_t = og_t \odot \sigma_c\,(c_t) \tag{4}$$

where "σ_c" denotes the "tanh" function used to compute the gate activation function:

$$\sigma\,(z) = (1 + e^{-z})^{-1} \tag{5}$$

The following formulas further describe the components of the time step "t" for the input gate, forget gate, cell candidate, and the output gate, respectively:

$$ig_t = \sigma_{cc}\big(Iw_{ig}\,i_t + Rw_{ig}\,o_{t-1} + b_{ig}\big) \tag{6}$$
$$fg_t = \sigma_{cc}\big(Iw_{fg}\,i_t + Rw_{fg}\,o_{t-1} + b_{fg}\big) \tag{7}$$
$$cc_t = \sigma_{cc}\big(Iw_{cc}\,i_t + Rw_{cc}\,o_{t-1} + b_{cc}\big) \tag{8}$$
$$og_t = \sigma_{cc}\big(Iw_{og}\,i_t + Rw_{og}\,o_{t-1} + b_{og}\big) \tag{9}$$

Similarly, LSTM has been used for wide range of text categorization applications, such as those applied in healthcare [22], settlement tweets [23], patents [24], hotel sentiment analysis [25], among others. LSTM has also been found to have better accuracy as compared to convolutional neural network (CNN), k-nearest neighbors (KNN), naïve bayes, among

others [24], [26], [27], [28]. Specifically, in one study, LSTM was found to performed better in small sample size if the number of hidden units and word embedding dimensions are set at both 50 [29].

SVM and LSTM are both popular due to their robust categorization performance, however, there seems to be limited studies that compared the performance of these two algorithms, particularly in categorizing text-based CS data written in natural languages. The closest studies available are found to be relatively of mix findings, wherein one is found to be better than the other and vice-versa. For instance, SVM has been proven to be more accurate than LSTM in categorizing type of music [30]. On the other hand, another study found that LSTM performed better than SVM in categorizing CS tweets of an Arabic telco [31]. Such mix findings are dependent on the complexity of textual data analyzed especially those with imbalanced distribution or overlap categories. Other variables that constitute on these issues include the size of datasets, feature engineering techniques used [18], and other factors. Nonetheless, the extent of evaluating the effectiveness of text categorization models in the context of CS still remain underexplored. Hence, in this paper, the author investigated the feasibility of designing text categorization models using SVM and LSTM algorithms in order to automatically categorize feedbacks, complaints, suggestions and recommendations. Specifically, the author analyzed the factors that affect the performance of these algorithms during training and implemented these algorithms in actual text categorization.

2. Methodology

In order to realize the main objectives of the study, the author utilized the following conceptual model, as shown on Fig. 4. The process started with the reading of the text data, followed by pre-processing, then the design of the text categorization models using SVM and LSTM algorithms, and the training and validation procedures. The final output of the study resulted to the actual implementation of the text categorization models during testing, with emphasis on the determination of machine learning performance of both algorithms. Each phase of the conceptual model is further detailed in the succeeding parts of the Methodology section.

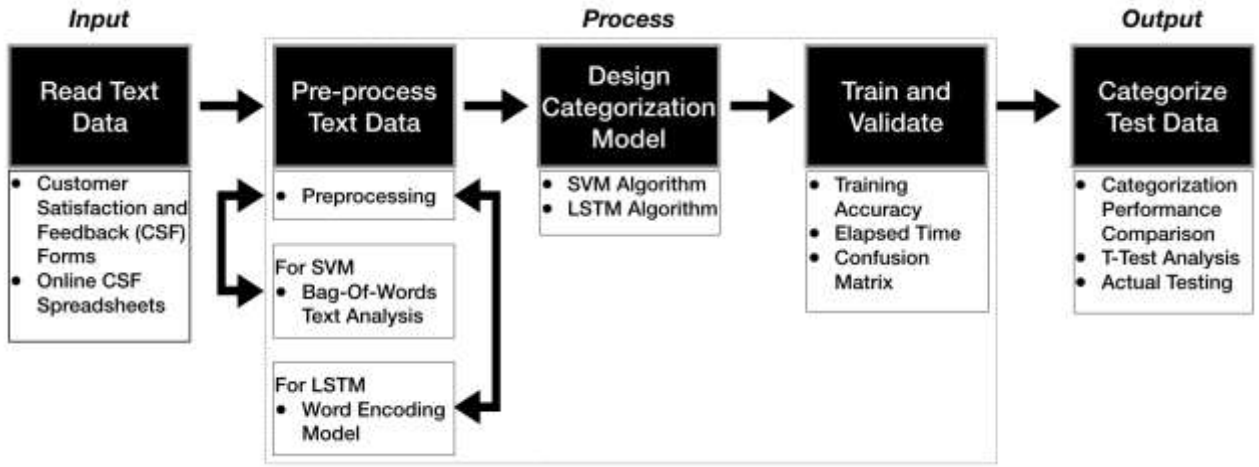


Fig. 4 - Conceptual Model

In the context of this study, the author utilized the text-based CS data, written in both English and Tagalog languages, from the Technological University of the Philippines in Manila, Philippines, as of August 30, 2020. Shown on Fig. 5 is a screenshot of the sample CS dataset maintained through online spreadsheets. The author set the "findings" (i.e. textual statements of CS) as input data, while the "categorization" (i.e. feedback, complaint, suggestion and commendation) as output data. In order to facilitate the preprocessing and modelling tasks, the author utilized MATLAB R2020a application in a computer workstation with specifications of 2.5 GHz CPU, 8GB 1600 MHz DDR3 RAM.

Findings	Categorization
Text	Categorical
Thank you for the courteousy of your office.	Commendation
The staffs are approachable and nice	Commendation
The office staff is very approachable	Commendation
Your office staff is approachable.	Commendation
Faster transactions please	Suggestion
Have a better performance	Commendation
I suggest to have a bigger rooms and student staff ti assst so that when it comes to getting an OJT...	Suggestion
I suggest feedback form should be compiled and be answered through ONLINE form. Less papers a...	Suggestion
I have nothing to say but a outstanding service we hope and the other student like me that you kee...	Commendation
The registrar asks me to pay for documentary stamps even there is no O.R issued. The front office ...	Complaint
I supposed to receive my TOR today but according to the staff on duty "There is no available paper...	Complaint
Government facility should not be having a 2nd and a HALF hours of break / lunch! nagdahilan pa ...	Complaint
May I suggest that for Certified True Copies. Asst. Registrar or any of its Director be given the auth...	Suggestion
Suggesting for instructions or guide to finish quickly the files needed.	Suggestion
Service is good. Keep it up. Thank you.	Commendation
Thank You so much for attending to our needs. Keep up your good works. God Bless.	Commendation
Thank you for all the years that I had experience here in three years of staying here. Napakasaya po.	Suggestion
Commendation for being friendly and corteous of the employee.	Commendation
The officer-in-charge was very attentive and friendly	Commendation
More personnel for efficient process.	Suggestion
Please continue annual job fair for newly grads.	Suggestion

Fig. 5 - Sample CS dataset

Eventually, the dataset has been divided into 4 trials wherein out of 1752 total datasets, 100%, 75%, 50% and 25% of it were randomly-selected in each trial. This was done to determine the effect of dataset size with accuracy and elapsed time during training. For each trial, the holdout percentage was set at 50% cross-validation threshold, and the MATLAB was reset during intervals. Likewise, the author pre-processed the input data in each trial in order to extract the most useful features intended for the subsequent text categorization models.

Shown on Fig. 6a is a screenshot of the function used to preprocess the input data. At this phase, the author converted each input data into tokenized words called tokens. The author then removed the stop words; normalized the words into their root-forms; converted them into lower cases; removed punctuation marks, words with less than 2 and/or more than 15 characters, HTML, XML and special characters. Meanwhile, Fig. 6b shows a sample result of preprocessing. In this particular example, a text originally-written as "*Thank you for the courtesy of your office"* has been trimmed down to *"thank courtesy office"* with a total of 3 tokens. Another one written as *"The staffs are approachable and nice"* has been reduced to *"staff approachable nice"* with also a total of 3 tokens.

preprocessText.mlx

```
function documents = preprocessText(textData)

% Tokenize the text.
documents = tokenizedDocument(textData);

% Remove a list of stop words then lemmatize the words. To improve
% lemmatization, first use addPartOfSpeechDetails.
documents = addPartOfSpeechDetails(documents);
documents = removeStopWords(documents);
documents = normalizeWords(documents,'Style','lemma');

% Erase punctuation.
documents = erasePunctuation(documents);

% Remove words with 2 or fewer characters, and words with 15 or more
% characters.
documents = removeShortWords(documents,2);
documents = removeLongWords(documents,15);

% Convert to lowercase.
documents = lower(documents);

end
```

Fig. 6a - Pre-processing function

Command Window

```
>> newText = "Thank you for the courtesy of your office";
newDocuments = preprocessText(newText)

newDocuments =

  tokenizedDocument:

    3 tokens: thank courtesy office

>> newText = "The staffs are approachable and nice";
newDocuments = preprocessText(newText)

newDocuments =

  tokenizedDocument:

    3 tokens: staff approachable nice
```

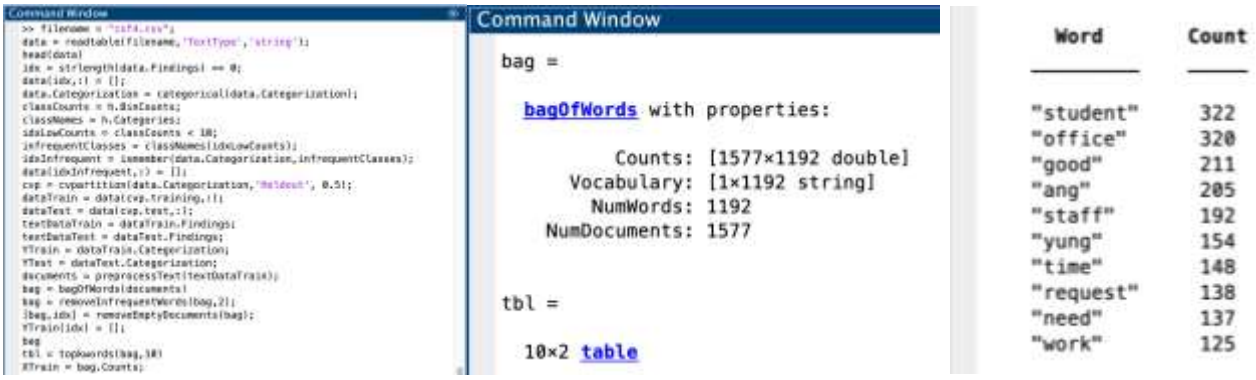
Fig. 6b -Pre-processing sample results

Furthermore, shown on Fig. 7 are the *word clouds* of both raw and cleaned words as the graphical representation of the results of pre-processing. Evidently, the cleaned data is more polished than the raw data since the unnecessary words and characters have been removed. The bigger words in the *word clouds* indicate that such words appear more frequently in the text analysis.

Fig. 7 - Word clouds

After the pre-processing, the author utilized feature engineering techniques to define the features needed for the subsequent text categorization models. In consideration on the compatibility constraints of the MATLAB software, the author initially used Bag-of-Words Model (BoWM), this time intended for the SVM categorization model. BoWM is a method of scoring the frequency on how many times a specific word has appeared in each text.

The following Fig. 8a, 8b and 8c show the BoWM function, its results, and most common tokens in the text analysis model, respectively. In this particular example, the BoWM generated a total of 1192 vocabularies or unique tokens out of total of 1752 datasets. The top 10 most commonly-known words generated are "student", "office", "good", "ang" (Tagalog of "the"), "staff", "yung" (Tagalog of "that"), "time", "request", "need" and "work". As noted, there are few Tagalog stop-words left unfiltered, wherein such issue seems attributable to the lack of multiple-language compatibility of the built-in functions of the simulation software.

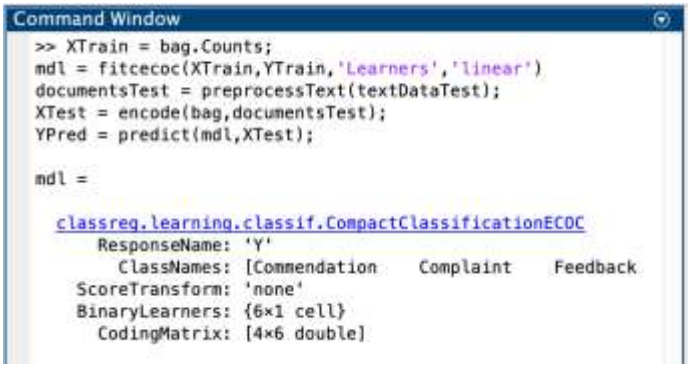


(a) BoWM Function **(b) BoWM Results** **(c) 10 Top Most Popular Words**

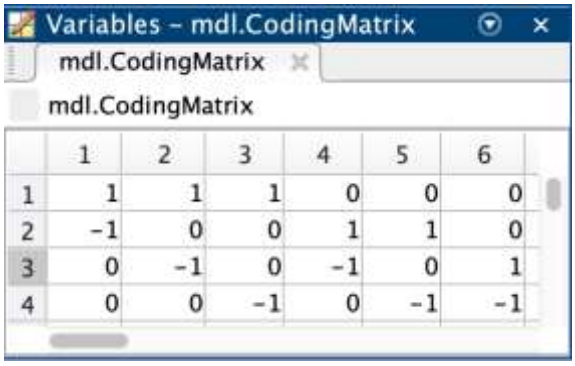
Fig. 8 - Bag-of-Words Text analysis model

Afterwards, the author then designed a SVM text categorization model using a compact, multi-category, Error Correcting Output Codes (ECOC) for SVM binary learners [32]. In this technique, the model utilizes a marginal binary learning algorithm using a coding matrix: $M \in \{+1, 0, -1\}^{cb}$ where "c" are the categories and "b" are the binary learners [33].

Shown on the following Fig. 9a is the function used to design and train the SVM text categorization model. For this purpose, the author utilized the tokenized documents of "findings", generated through the BoWM, as predictors and "categorization" as linear response. Meanwhile, Fig. 9b shows the 4x6 coding matrix used to map out how the binary learners, which in this case made of 6 best soft-margins, train in the 4 categories, which are composed of 1=commendation; 2=complaint; 3=feedback; 4=suggestion. The elements in the coding matrix correspond to the prediction results of the 6 binary learners and 4 categories. Each element in the matrix is expressed as either "-1" which means that "b" allocates the predictions into a negative category; "0" which means that "b" removes predictions from the dataset before training; and "+1" which means that "b" allocates predictions to a positive category.



(a) Compact Multi-Category Function



(b) 4x6 Coding Matrix

Fig. 9 - SVM Text categorization model design

Moreover, the following Table 1 shows the specifications of the binary learners used in the design of the SVM categorization model composed of prior probability, cost, beta, bias, lambda, regularization and learning weights. On the other hand, Table 2 details out the equivalent scores of binary learners in categorizing sample tokenized documents.

Table 1 - SVM Binary Learners Specifications

Property	B1	B2	B3	B4	B5	B6
Category Name	[-1, 1]	[-1, 1]	[-1, 1]	[-1, 1]	[-1, 1]	[-1, 1]
Prior Probability	[0.1556, 0.8444]	[0.2762, 0.7238]	[0.2597, 0.7403]	[0.6744, 0.3256]	[0.6557, 0.3443]	[0.4790, 0.5210]
Cost	[0,1;1,0]	[0,1;1,0]	[0,1;1,0]	[0,1;1,0]	[0,1;1,0]	[0,1;1,0]
Beta	1192 B_1	1192 B_2	1192 B_3	1192 B_4	1192 B_5	1192 B_6
Bias	0.8624	0.6407	0.4019	-0.4115	-0.6283	-0.2685
Lambda	0.0019	0.0016	.0016	0.0039	0.0041	0.0030
Regularization	'ridge (L2)'	'ridge (L2)'	'ridge (L2)'	'ridge (L2)'	'ridge (L2)'	'ridge (L2)'
Learner Weights	0.6178	0.7208	0.7048	0.2952	0.2792	0.3822

Table 2 - SVM Binary Learners' Token-Scores

Token	B1	B2	B3	B4	B5	B6
nbc	-0.200443084	-0.152848724	0	0.078796828	0.16039847	0.119696682
cycle	0	-0.152848724	0	-0.076535546	0	0.119696682
evaluation	0	-0.152848724	0	-0.076535546	0	0.119696682
verify	0	-0.152848724	0	-0.076535546	0	0.119696682
letter	0	-0.152848724	0	-0.076535546	0	0.119696682
appeal	0	-0.685123312	0	-0.22288367	0	0.434609923
promotion	0.025470719	-0.05125248	0.071357193	-0.076535546	0	0.119696682
additional	0	-0.4178241	-0.331097529	-0.144222098	-0.148941087	0.091687468
licensure	0	-0.315560091	0	-0.110199807	0	0.293513934
exam	0	-0.315560091	0	-0.110199807	0	0.293513934
university	-0.023976476	-0.158732993	0.099206511	-0.089929039	0.033844765	0.267281832
extension	-0.040194664	0	-0.023560592	0.043799086	0.000198724	-0.006663604
services	-0.040194664	0	0	0.043799086	0.033844765	0
file	-0.040194664	0.024948214	0.296829349	0.001094384	0.021676626	0.012090331
complaint	-0.438951387	-0.03948172	-0.228991836	0.326753534	0.51250204	-0.000616271
against	-0.040194664	0	0	0.043799086	0.033844765	0

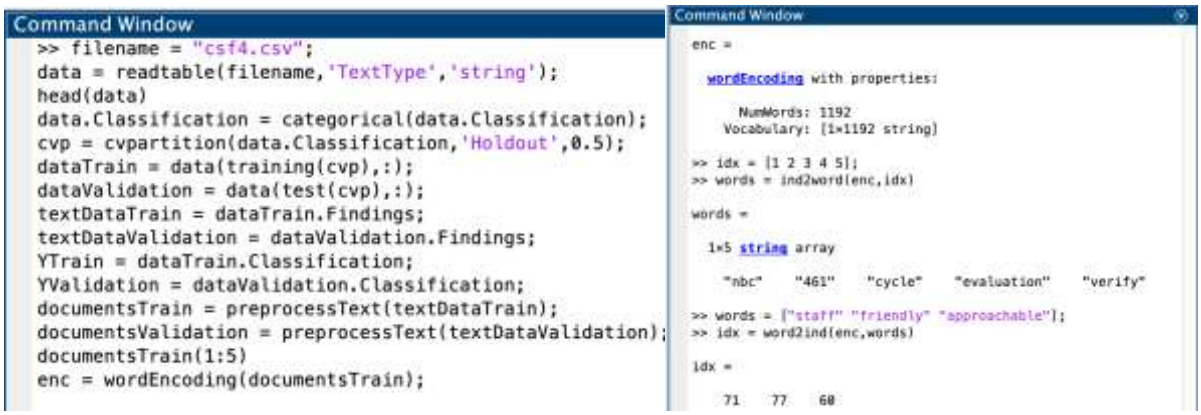
recognize	-0.040194664	0	0	0.043799086	0.033844765	0
excess	-0.040194664	0	0	0.043799086	0.033844765	0
baggage	-0.040194664	0	0	0.043799086	0.033844765	0

The performance of the binary learners is measured in terms of categorization function, expressed as [34], [35]:

$$\hat{c} = \underset{c}{argmin} \frac{\sum_{b=1}^{B} |m_{cb}| \, l\,(m_{cb}, s_b)}{\sum_{b=1}^{B} |m_{cb}|} \tag{10}$$

where "ĉ" is the predicted category of the observation used to minimize the total losses of binary learners "B"; "m_{cb}" is an element of coding matrix "M" or the code equivalent to category "c" of the binary learner "b"; "s_b" is the predicted score of "b"; and "l" is the binary loss function.

Meanwhile, another feature engineering technique, called the *word encoding* (WE), was implemented, this time, for the LSTM. WE is a word representation model which translates tokenized documents into sequences of numeric indices and back. Shown on Fig. 9a is a screenshot of the actual WE function, which generated a total of 1192 vocabularies or unique tokens. Fig. 9b, on the other hand, elaborates a sample representation of how sample tokens were encoded sequentially. The first example shows that using WE model, the equivalent encoded words of 1, 2, 3, 4, and 5 indices are "nbc, 461, cycle, evaluation and verify", respectively. The second example, on the other hand, shows the equivalent indices of the previous tokenized documents sampled in Fig. 5b, read as *"staff friendly approachable"* with indices of "71, 77, 60", respectively.

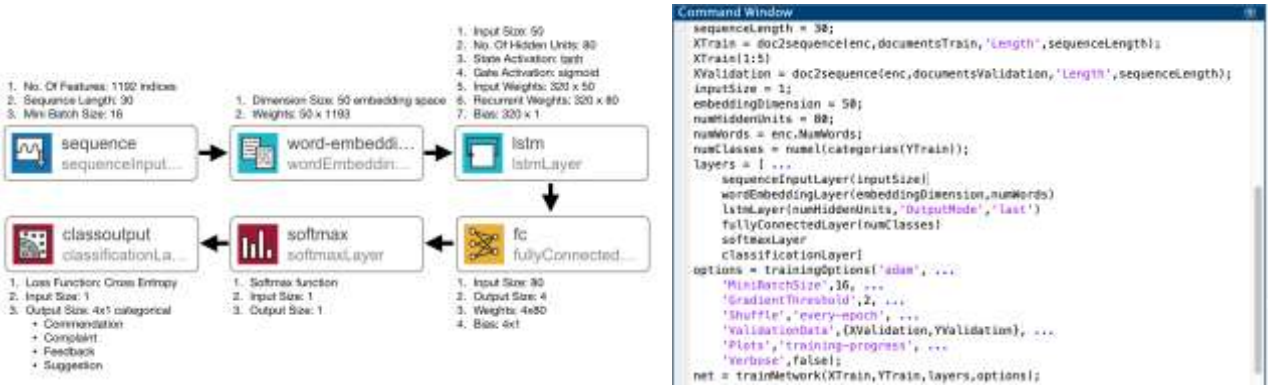


(a) WE Function **(b) Sample Encoding Results**

Fig. 10 - Word encoding text analysis model

The extent of converting tokenized words into indices using WE model is to ensure that the input data are sequentially arranged first prior to LSTM modelling. This technique is required in an LSTM in order to address the vanishing gradients caused by sparse and highly dimensionality of text data. These sequences of numeric indices were then further optimized in the subsequent *word embedding* (W-E) layer of the LSTM text categorization model. The author set the sequence length of the vocabularies into "30" since majority of them have 30 tokens. The author then left-padded the words with lesser than 30 and truncated those greater than it. This technique was employed for both training and validation data, wherein the held-out was also set at 50% cross validation threshold.

Illustrated on Fig. 11a and 11b are the architecture and actual function used to design the LSTM text categorization model, respectively. Specifically, the author set a 1 sequence input layer composed of 1192 +1 indices of vocabularies or features; 1 W-E layer, which maps out the input vocabulary words into real output vectors and captures their meanings and relationships with other words. Each of these 1193 indices has 50 embedding dimensions or vocabulary size, 30 sequence length and 16 mini batch size set during training. Moreover, the author designed a LSTM layer with 80 hidden units; then 4 fully connected layers; 1 softmax layer based on softmax function; and 1 categorization (or classification) based on cross-entropy function. Afterwards, the author trained the network at max of 30 epochs, using adaptive moment estimation (Adam) solver [36] with 2 gradient thresholds and 0.01 initial learning rate.



(a) Architecture **(b) Function**

Fig. 11 - LSTM Text categorization model design

Fig. 12 depicts the sample weights of the inputs, recurrent, and bias, used by the LSTM layers. These data are crucial toward the effective functioning of the subsequent fully connected, softmax and output layers.

Iw			***Rw***			***B***
320 x 50 Input Weight			**320 x 80 Recurrent Weight**			**320x1 Bias Weight**
-0.010505933	.....	0.006799366	-0.012307452	.....	-0.04799195	0.067843631
-0.006725231	.....	-0.001689665	-0.022708198	.....	-0.03300222	0.040509921
-0.016484963	.....	0.008511145	-0.065878816	.....	-0.069770545	0.031176792
-0.011134897	.....	-0.006692105	0.043638799	.....	-0.001454611	0.035273284
-0.006088676	.....	0.005235719	0.016654223	.....	0.000595269	0.021964159
.		.	.		.	.
.		.	.		.	.
.		.	.		.	.
-0.020269442	.....	0.021786401	-0.018277168	.....	0.001239109	0.049391158
-0.003194538	.....	-0.002451337	0.032106604	.....	-0.008613584	0.10798635
-0.019763082	.....	0.010774263	-0.060540121	.....	-0.12896396	0.029976076
0.065526344	.....	-0.054802474	0.05173564	.....	-0.010183136	0.081644848
-0.02989902	.....	0.033475418	-0.008851284	.....	-0.028676083	0.024114709

Fig. 12 – Input, Recurrent and Bias Weights of 50 Dimensions x 1193 Tokens

The categorization performance of the LSTM output layer was computed using the following cross entropy loss formula [37]:

$$CE = -\sum_{a=1}^{D}\sum_{b=1}^{C} x_{ab} \ln z_{ab}, \tag{2}$$

where "D" is the no. of datasets; "C" is the no. of categories; "x_{ab}" signifies that "a_{th}" dataset is part of the "b_{th}" category; "z_{ab}" is the output of dataset "a" for category "b" which is equivalent to the probability where the networks correlates "a_{th}" with "b_{th}", also known as the softmax function.

Afterwards, the author evaluated the performance of both SVM and LSTM algorithms during training, wherein initially, the categorization accuracy of each model was determined by the following formula [14]:

$$CA = \frac{(Pt+Nt)}{(Pt+Pf+Nf+Nt)} \tag{3}$$

where "CA" is the categorization accuracy; "Pt" is no. of true positives; "Nt" is no. of true negatives; "Pf" is no. of false positives; and "Nf" is no. of false negatives. Likewise, the confusion error rating of both models was determined using the following formula [38]:

$$Ce = \frac{\sum_{o=1}^{n} w_o k_o}{\sum_{o=1}^{n} w_o} \tag{4}$$

where "w_o" is the weight of observation "o"; "k_o=1" if the predicted category of "o" differs from its true class, otherwise it is "0". Aside from the training accuracy performance, the elapsed time of both models were also recorded based on the results of MATLAB simulation. Subsequently, the author compared the overall training performance in terms of accuracy and elapsed time of both SVM and LSTM models using the following independent T-test formula [39]:

$$t = \frac{\bar{s} - \bar{l}}{\sqrt{\left(\frac{s_d^2}{n1} + \frac{l_d^2}{n2}\right)}} \tag{5}$$

where "$\bar{s}$" is SVM total dataset average; "$\bar{l}$" is the LSTM total dataset average; "s" is SVM sample dataset size; "l" is LSTM sample dataset size; "s_d" is SVM standard deviation (SD); and "l_d" LSTM SD. The calculated "t" value was compared with the critical t-value distribution table, in which the degrees of freedom "f" was computed as follows:

$$f = \frac{\left(\frac{s_d^2}{s} + \frac{l_d^2}{l}\right)^2}{\frac{1}{s-1}\ (\frac{s_d^2}{s})^2} \tag{6}$$

3. Results and Discussion

The performance of both SVM and LSTM text categorization models during training were evaluated in terms of accuracy, confusion error and elapsed time. As reflected on Table 3, the results of performance evaluation for both SVM and LSTM models affirm that while the number of dataset size is increased, the learning accuracy is likewise improved. Inevitably, the confusion rating decreases and the training elapsed time slows down.

Table 3 – SVM and LSTM Training Performance

Dataset Size	SVM			LSTM		
	Accuracy (%)	Confusion error (%)	Elapsed Time (s)	Accuracy (%)	Confusion error (%)	Elapsed Time (s)
1752	98.63	1.4	3.78	99.32	0.70	84
876	92.01	8.0	2.53	96.18	3.80	35
438	82.65	17.4	2.25	87.69	12.30	18
219	56.88	43.10	1.56	62.50	37.50	15
$\bar{x}$	82.54	17.48	2.53	86.42	13.58	38
SD	18.32	18.30	0.93	16.69	16.69	31.91

Notes: $\bar{x}$ = average; SD = standard deviation

Fig. 13 shows the confusion matrix of SVM categorization model according to its best performance during training. As shown, SVM model was able to generate 98.6% accuracy, 1.4% confusion error rating, and completed the training at 3.78s. Overall, the model categorized the training data correctly except for some commendations that were miscategorized as either complaints (0.7%) or suggestions (0.7%).

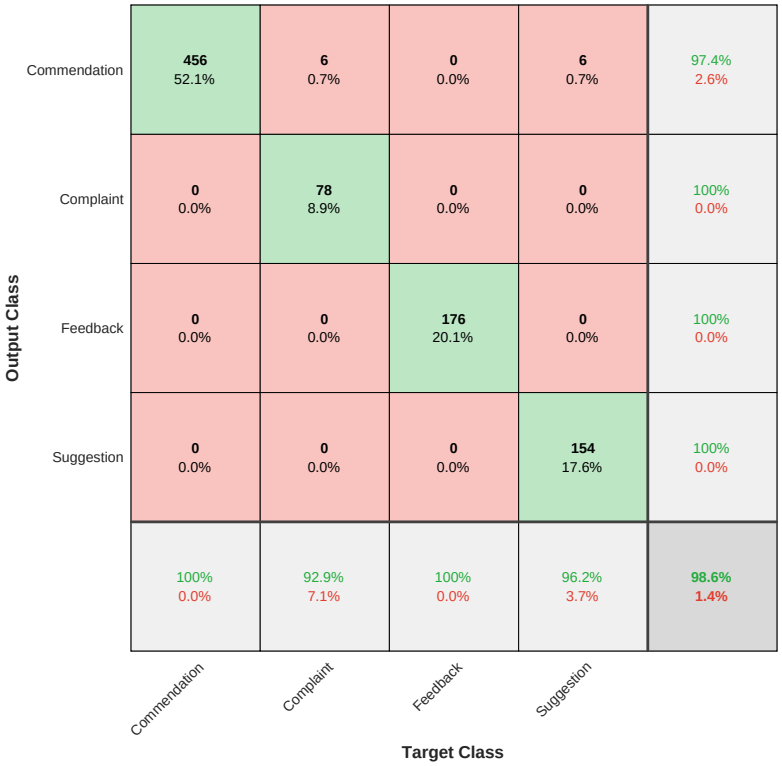


Fig. 13 - SVM Model Confusion Matrix

On the other hand, the following Fig. 14 shows the confusion matrix of LSTM categorization model. As shown, its best performance was recorded at 99.32% accuracy with 0.70% confusion error rating, and elapsed time of 84s. While the model performed well overall, there were 6 instances when the commendations were miscategorized as feedbacks (0.7%).

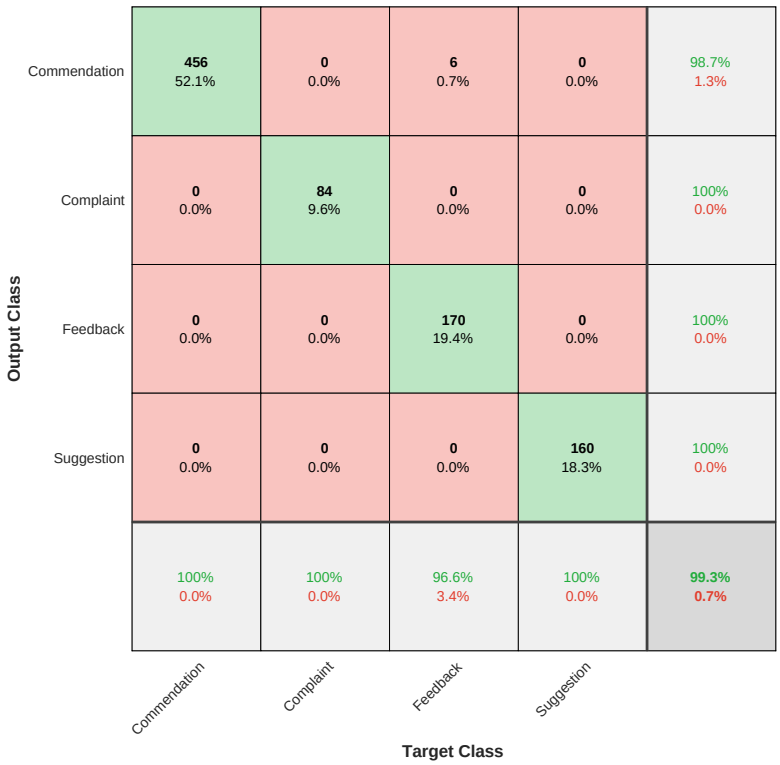


Fig. 14 – LSTM Model Confusion Matrix

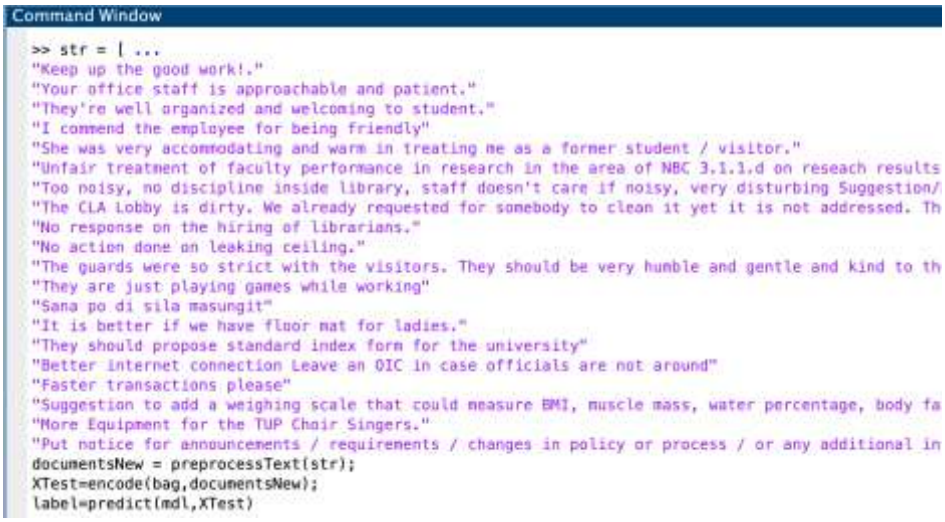
Meanwhile, Table 4 elaborates the results of independent t-test conducted to inferentially compare the performance of the two algorithms in terms of accuracy and elapsed time during training.

Table 4 - Independent Samples T-Test Results

	Levene's Test of Equality of Variances		t-test for Equality of Means		Sig. (2-tailed)	Mean Diff	Std. Error Diff.	95% Confidence Interval of the Difference	
	F	Sig	t	df				Lower	Upper
Accuracy Equal Variances Assumed	0.015	0.907	0.313	6	0.765	3.88	2.391	-26.44	34.2
Accuracy Equal Variances Not Assumed			0.313	5.948	0.765	0.88	2.391	-25.504	34.264
Elapsed Time Equal Variances Assumed	0.398	0.045	2.222	6	0.068	35.47	15.96	-3.582	74.522
Elapsed Time Equal Variances Not Assumed			2.222	3.005	0.113	35.47	15.96	-15.273	86.213

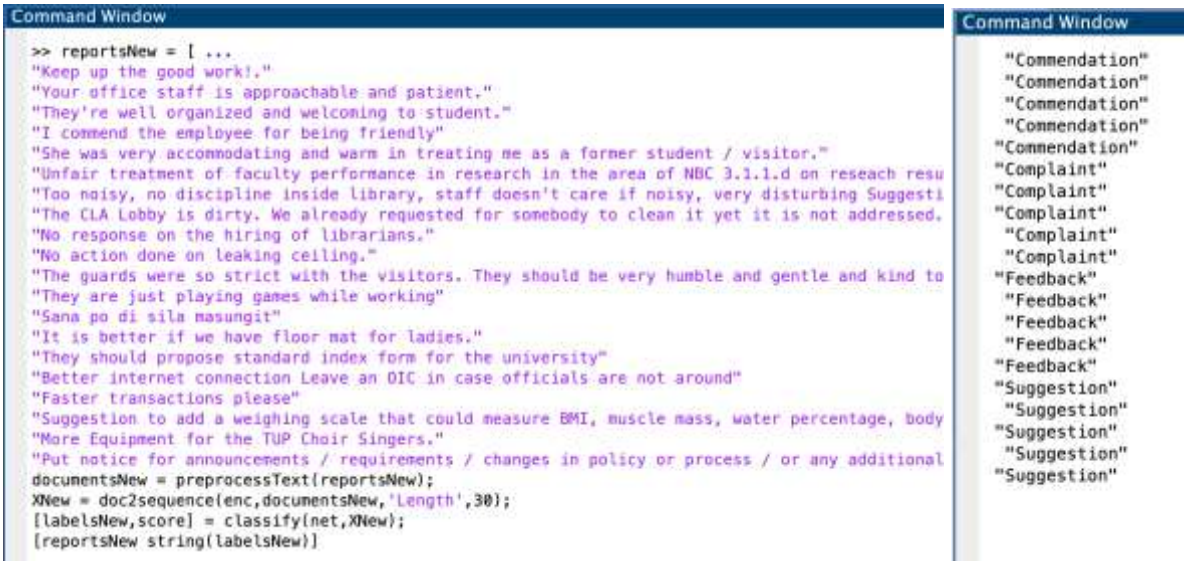
In terms of accuracy, although the LSTM has recorded a higher average rating of 86.42% or best rating of 99.32%, such results are not statistically significantly different with that of the SVM, which has an average rating of 82.54% or best rating of 98.63%. This finding is corroborated by the statistical values of t (5.948) =.313 and p=.907. Conversely, in terms of training elapsed time, the study found that SVM is statistically significantly faster (2.53 ± .93s) than LSTM (38.00 ± 31.91s), with t (3.005) = 2.222 and p= 0.045. Hence, these results mean that both SVM and LSTM algorithms are at par with each other in terms of training accuracy, but SVM is found to be significantly faster than LSTM by approximately 35.47s.

After training, the author deployed the best SVM and LSTM categorization models in order to validate their respective training performances. Specifically, the author utilized 20 randomly-selected test dataset composed of 5 feedbacks, 5 commendations, 5 suggestions and 5 complaints. The results of testing are shown on Figure 15 and 16 for SVM and LSTM categorization models, respectively. Interestingly, both models have predicted 100% of all the test data. This means that the design process conducted was effective in implementing SVM and LSTM algorithms toward actual text-based CS categorization application.



(a) SVM Test Input Data **(b) SVM Test Output Data**

Fig. 15 - SVM Actual test categorization Results



(a) LSTM Input Test Data **b) LSTM Output Test Data**

Fig. 16 - LSTM Actual test categorization results

4. Conclusions

Although LSTM has higher training accuracy than SVM in most trials conducted, their difference was not statistically significant. Conversely, SVM was found to be statistically significantly faster than LSTM. Such findings could be attributed on several factors such as the limitations of the dataset size, high-dimensionality of both English and Tagalog languages analyzed, and the applicability of the feature engineering techniques used. Interestingly, both models were found to be 100% accurate in predicting the test data during actual implementation. This means that the modelling approach conducted was effective and feasible in establishing the categorization capabilities of both SVM and LSTM algorithms as applied to text-based CS data. Hence, the choice of using between the two algorithms boils down on the extent of available dataset and the technical skills in using appropriate feature engineering techniques and in deploying these algorithms toward actual text categorization applications.

Future studies should consider standardizing a common feature engineering technique applicable for both algorithms. Likewise, when using multiple languages, consider translating non-English text, such as Tagalog, into pure English texts prior to modelling. This is to ensure that certain stop words written in the local language are accurately lemmatized and pre-processed. Likewise, the effect of modifying the holdout percentage, embedding dimension (for the case for LSTM) and adding more CS input data, particularly thousands of samples to essentially establish deep learning, should also be explored further.

Acknowledgement

The author would like to thank the continued support of the top management of the Technological University of the Philippines - Manila, particularly in upholding customer satisfaction, which is an essential component of the University QMS. Likewise, gratitude is given to the staff of the quality assurance office – namely, Deseree Joy D. Lingal, Mark Joseph O. Indelible, Aileen T. Dollisen, RJ P. Icatar, and Ronnie F. Ladores Sr. – in facilitating various QMS activities, particularly, in gathering, analyzing and reporting CS data, timely and efficiently.

References

[1] International Organization for Standardization. 2015. ISO 9001:2015 - Quality management systems – Requirements. Retrieved from https://www.iso.org/standard/62085.html

[2] Corpuz, R.S.A. (2019). Implementation of artificial neural network using scaled conjugate gradient in ISO 9001:2015 audit findings classification. International Journal of Recent Technology and Engineering, 8 (2), 420-425

[3] International Organization for Standardization. 2015. ISO 9000:2015 - Quality management systems - Fundamentals and vocabulary. Retrieved from https://www.iso.org/standard/45481.html

[4] International Organization for Standardization. 2018. ISO 10004:2018 - Quality management – Customer Satisfaction – Guidelines for monitoring and measuring. Retrieved from https://www.iso.org/standard/71582.html

[5] Lessman, S. & Vob, S. (2009). A reference model for customer-centric data mining with support vector machines. European Journal of Operational Research, 199, 520-530.

[6] Zablith, F. & Osman, I. (2019). ReviewModus: Text classification and sentiment prediction of unstructured reviews using a hybrid combination of machine learning and evaluation models. Applied Mathematical Modelling, 71, 569-5.
[7] Al-Smadi, M., Qawasmeh, O., Al-Ayoyoub, M., Jararweh, Y. & Gupta, B. (2018). Deep recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotel's reviews. Journal of Computational Science 27, 386-393.
[8] Sebastiani, F. (2002). Machine learning in automated text categorization. ACM Computing Surveys, 34, 1, 1-47.
[9] Qing, L., Linhong, W. and Xuehai, D. (2019). A novel neural network-based method for medical text classification. Future Internet, 11 (255), 1-13.
[10] Gopalakrishnan, V. & Ramaswamy, C. (2017). Patient opinion mining to analyze drugs satisfaction using supervised learning. Journal of Applied Research and Technology, 15 (4), 311-319.
[11] Vapnik, V. (1999). The Nature of Statistical Learning Theory. Second edition. Springer: Berlin, Germany.
[12] Coussement, K. & Van den Poel, D. (2008). Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert Systems with Applications, 34 (1), 313-327.
[13] Yin, C., Xiang, J., Zhang, H., Wang, J., Yin, Z. & Kim, J-U. (2015). A new SVM method for short text classification based on semi-supervised learning. Pages 100-103 in IEEE, editors, 4th International Conference on Advanced Information Technology and Sensor Application (AITS), 21-23 August 2015, Harbin, China.
[14] Corpuz, R.S.A. (2020). ISO 9001:2015 risk-based thinking. Makara Journal of Technology, 24 (3), 149-159.
[15] Christianini, N. & Shawe-Taylor, J. (2000). An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press: Cambridge, UK.
[16] Christianini, N. & Schölkopf, B. (2002). Support Vector Machines and Kernel Methods: The New Generation of Learning Machines. AI Magazine, 23 (3), 31-41.
[17] Lo, S. (2008). Web service quality control based on text mining using support vector machine. Expert Systems with Applications, 34, 603-610.
[18] Raza, M., Hussain, F. K., Hussain, O. K., Zhao, M. & Rehman, Z. (2019). A comparative analysis of machine learning models for quality pillar assessment of SaaS services by multi-class text classification of users' reviews. Future Generation Computer Systems, 101, 341-371.
[19] Cheung, K.-W., Kwok, J.T., Law, M.H., & Tsui, K.-C. 2003. Mining customer product ratings for personalized marketing. Decision Support Systems, 35, 231-243.
[20] Lee, C-H. & Yang, H-C. (2005). A classifier-based text mining approach for evaluating semantic relatedness using support vector machines, DOI: 10.1109/ITCC.2005.2 in IEEE, editors. Proceedings, International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II, 4-6 April 2005, Las Vegas, NV, USA.
[21] Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9 (8), 1735-1780.
[22] Hu, Y., Wen, G., Ma, J., Wang, C., Lu, H. & Huan, E. (2018). Label-indicator morpheme growth on LSTM for Chinese healthcare question department classification. Journal of Biomedical Informatics 82: 154-168.
[23] Huang, R., Taubenbock, H., Mou, L. & Zhu, X.X. (2008). Classification of settlement types from tweets using LDA and LSTM. Pages 6408-6411 in IEEE, editors. Proceedings, 2018 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), 22-27 July 2018, Valencia, Spain.
[24] Xiao, L., Wang, G. & Zuo, Y. (2018). Research on patent text classification based on word2vec and LSTM. Pages 71-74 in IEEE, editors. Proceedings, 11th International Symposium on Computational Intelligence and Design (ISCID), 8-9 December 2018, Hangzhou, China.
[25] Khotimah, D.A.K. & Sarno, R. (2019). Sentiment analysis of hotel aspect using probabilistic latent semantic analysis, word embedding and LSTM. International Journal of Intelligent Engineering and Systems, 12 (4), 275-290.
[26] Luan, Y. & Lin, S. (2019). Research on Text Classification Based on CNN and LSTM. Pages, 352-354 in IEEE, editors. Proceedings, 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 29-31 March 2019, Dalian, China.
[27] Wang, J., Liu, T-W., Luo, X. & Wang, L. (2018). An LSTM approach to short text sentiment classification with word embeddings. Pages 214-223 in IEEE, editors. Proceedings, 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018), October 2018, Hsinchu, Taiwan.
[28] Gharibshah, Z., Zhu, X., Hainline, A. & Conway, M. (2020). Deep learning for user interest and response prediction in online display advertising. Data Science and Engineering, 5 (26), 12-26
[29] Corpuz, R.S.A. (2021). An application method of long short-term memory neural network in classifying english and tagalog-based customer complaints, feedbacks, and commendations. International Journal on Information Technologies and Security, 13 (1), 2021, pp. 89-100.
[30] Fulzele, P., Singh, R., Kaushik, N. & Pandey, K. (2018). A hybrid model for music genre classification using LSTM and SVM. DOI: 10.1109/IC3.2018.8530557 in IEEE, editors. Proceedings, Eleventh International Conference on Contemporary Computing (IC3), 2-4 August 2018, Noida, India.

[31] Almuqren, L.A.R., Qasem, M.M. & Cristea, A.I. (2019). Using deep learning networks to predict telecom company customer satisfaction based on Arabic tweets. Paper presentation, 28th International Conference on Information Systems Development (ISD), August 28-30, 2019, Toulon, France.
[32] Dietterich, T. G. and G. Bakiri (1995). Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research, 2, 263–286.
[33] Allwein, E. L., R. E. Schapire, and Y. Singer (2000). Reducing multiclass to binary: a unifying approach for margin classifiers. Journal of Machine Learning Research, 1, 113–141.
[34] Fürnkranz, J. (2002). Round Robin Classification. Journal of Machine Learning Research, 2, 721–747.
[35] Escalera, S., Pujol, O. & Radeva, P. (2010). On the decoding process in ternary error-correcting output codes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32 (7), 120–134.
[36] Kingma, D. & Ba, J. (2015). Adam: A method for stochastic optimization. Retrieved from https://arxiv.org/abs/1412.6980
[37] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer: New York, NY, USA.
[38] MathWorks Inc. (2019). Classification ECOC Loss. Retrieved from https://www.mathworks.com/help/stats/classificationecoc.html.
[39] Villanueva, A.B. & Corpuz, R.S.A. (2020). Design and development of fire evacuation system using fuzzy logic control. International Journal of Scientific and Technology Research, 9(4), 2096-2103.