IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 30, 2020, accepted August 30, 2020, date of publication September 30, 2020, date of current version October 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3027845

Sentiment Analysis by Fusing Text and Location Features of Geo-Tagged Tweets

WEI LUN LIM, CHIUNG CHING HO, (Senior Member, IEEE), AND CHOO-YEE TING
Faculty of Computing and Informatics, Multimedia University, Cyberjaya 63100, Malaysia

Corresponding author: Chiung Ching Ho (peyter@ieee.org)

This work was supported by Telekom Malaysia Research and Development under Grant MMUE/180019.

ABSTRACT Twitter sentiment analysis provides valuable feedback from public emotion concerning certain events or products. Current research has been focused on obtaining sentiment features from vectorized lexical and syntactic feature from tweets, without further context. In this paper, we demonstrated how vectorized location information could be combined with word embeddings to produce a hybrid representation, which has resulted in an improvement on a tweet sentiment classification task. The location information of the geo-tagged tweets provided further context, which was useful for a sentiment classification task. The tweets investigated contained a set of geo-tagged tweets. The word embeddings of these tweets were combined with the geo-tagged tweets' vectorized location features to form a sentiment feature set of geo-tagged tweets. The sentiment feature set was incorporated into a convolution neural network and a bi-directional long short-term memory network for the tasks of training and predicting of sentiment classification labels. This hybrid representation is compared with the baseline GloVe model through a few experiments, and the results have shown that the incorporation of vectorized location information has resulted in improvement of the accuracy for the twitter sentiment classification task.

INDEX TERMS Deep learning, geo-tagged tweet, information representation, location intelligence, multi-modal fusion, pattern recognition, sentiment classification.

I. INTRODUCTION

Twitter is the standard micro-blog platform that has been analysed by researchers due to the large user base of over 319 million active users [1]. The short limits of words (280 characters) for tweets allows for the concise expression of objective and opinionated content. Despite adding more disambiguation, the short sentences in tweets have eased the analytic process for sentiment analysis, especially if the context to the tweet is incorporated. Geo-tagged tweets contain location information which has added context to the content of the tweet. Research has shown that sentiment classification accuracy that uses location as a feature has outperformed sentiment classification, which only uses text [2].

Sentiment analysis is a general term while the most popular task being polarity detection [3]. The term polarity detection and sentiment analysis are used interchangeably due to the limited definition of sentiment analysis as the Natural Language Processing (NLP) task that categorizes a text as being positive or negative.

Recent approaches to polarity detection were focused on deep learning neural networks with word vector representations [4]. Deep learning has gained popularity for NLP tasks due to the convenience of automated feature extraction. In traditional NLP approaches, lexical and syntactic features of the text are specified explicitly. In contrast, automatic feature representation and extraction are performed using deep learning techniques perform, and this approach has often led to a better outcome [5].

In this article, the convolution neural network (CNN) and bidirectional long short-term memory (BiLSTM) network were used to perform sentiment classification on a geo-tweet Twitter dataset. Geo-tagged tweet is a tweet containing geographic coordinate (latitude, longitude) that indicates the location where the tweet was generated.

The global vectors for word representation [6] were first used to transform the text into word embeddings. Subsequently, the category of locations which are nearby each geo-tagged tweet is treated as being representative of the tweet's

The associate editor coordinating the review of this manuscript and approving it for publication was Shiqing Zhang.

location information. This approach is inspired by Firth's assertion that ''you shall know a word by the company it keeps'' - and inspired us to consider that perhaps we can know a geo-tagged tweet by the company it keeps as well [7].

The categories of nearby locations for each geo-tagged tweet was vectorized using one-hot encoding and frequency-count encoding. The nearby location for each geo-tagged tweet was also vectorized as an ego network. The word embeddings subsequently were then concatenated with the vectorized location and fed into a CNN and a BiLSTM network to train and classify the sentiment labels of the tweet. The experimental results have shown that our approach has resulted in improved classification performance when compared to using word embeddings alone.

The rest of the paper is organized as follows. In Section II, a review of related work is shown. Section III discusses the geo-tagged twitter dataset which was used in this study. Section IV presents the methodology used for deep learning twitter text classification. Section V describes the architecture of the CNN and BiLSTM models with concatenated word embeddings and vectorized location features. Section VI describes the experiments performed in this study. Section VII discusses the experiment results. Finally, conclusions and future works are presented in Section VIII.

II. RELATED WORK

In this section, a review of related works is performed. The motivation for performing this study is presented, followed by a discussion on twitter text classification, as well as word embeddings models and deep learning models used for sentiment analysis.

A. LOCATION AND SENTIMENT

There is a fundamental link between feelings and space: locations have an aura that can elicit powerful and varied emotions in people. Places able to trigger feelings such as boredom, attraction, relaxation, frightening or threatening. The loss of a place do affect emotional feelings [8].

A study of sentiment in New York City [9] have shown that public sentiment is most positive at public parks and is most negative at the transportation hub. Cemeteries, medical centres, jails, and sewage facilities are shown to be associated with a strong sentiment as well. Public opinion can vary across fine-grained locations in public places, where locations with significant litter or exposure to a travelling community were associated with strong negative sentiments. At the same time, well-kept natural surroundings elicited strong positive sentiments [10].

There is a correlation between urban metrics and sentiment analysis. Generally, people had the opinion that shrinking cities exhibited more negative sentiment as compared to growing cities, however, in reality the pattern of sentiment exhibited in both shrinking and growing cities are the same [11]. Loss in population did not contribute to negative sentiment, as this was attributed to poor infrastructure. In another study, the sentiments exhibited by different income-level groups were studied. By analyzing urban outdoor images, the high-income group generated more positive sentiment. In contrast, medium-income and low-income group generated more negative sentiment [12]. In another study, tweets related to city features and transportation activities were analyzed to create a polarity city features map [13]. Through this map, the travellers can avoid the areas associated with negative sentiments, thus improving the satisfaction level of travellers.

Spatial correlation with sentiments, especially for positive sentiments, is also demonstrated in research performed using the United States Geo-tagged data [14]. The evidence for relating location to sentiment can be shown even on a country-level, with research showing that different areas in the same country have different sentiment towards the same topic [15], [16].

Investigation on the impact of location information on sentiment classification can result in better decision support. Research has shown that sentiment indications will improve performance of POI recommendation [17]. Rating of a business and rating of its geographical neighbourhood have a weak positive correlation. The incorporation of geographical neighbourhood influence like category of the business, its popularity, and the content of review from customers can further improve the business rating prediction accuracy. The geographical distance between a business and a user adversely affects the prediction accuracy in point of interest (POI) recommendation and prediction [18].

From the literature reviewed, there is strong evidence that sentiment can be affected by location or spatial factors. While spatial and temporal analysis of geo-tagged tweets has been performed, there has been little work which has explored the incorporation of location information as a feature for performing geo-tagged tweet sentiment classification.

B. TWITTER TEXT CLASSIFICATION

Sentiment analysis is treated as a text classification problem that classifies an opinionated document as expressing a positive or negative opinion which aims to find the general sentiment of the author in an opinionated text [19]. The text classification process is composed of the following elements: (1) data acquisition, (2) data analysis and labelling, (3) feature construction and weighting, (4) feature selection (5) model training, and (6) solution evaluation [20]. Fig 1 shows a text classification process workflow. The text representation and classifier components of the process is discussed in Sections C and D.

Most of the past research work performed on social media analysis research has been performed using Twitter data [21]. This could be due to Twitter's social network rules, where any user can follow any other user as well as access to its data. Twitter's application programming interface (API) has provided access to almost 100% of its data, which is an access rate unrivalled by other social network platforms [22].

As a result of this, Twitter sentiment analysis is applied in many research to improve the understanding of people's

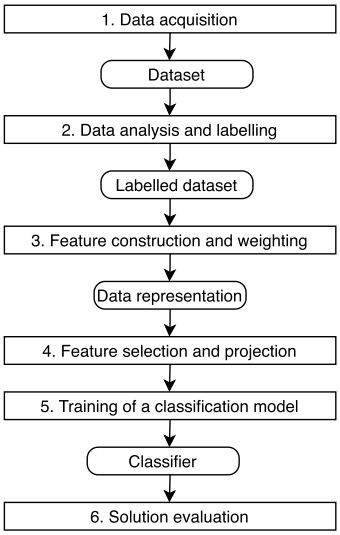


FIGURE 1. Text classification process with state-of-the-art element [20].

TABLE 1. List of supervised machine learning algorithms and how they work.

Rule-based use IF-THEN rules to perform classification
Decision tree uses a tree-like graph structure to perform classification
SVM finds a separation between hyperplanes of data class
ANN attempts to imitate human neurons to perform classification
Deep learning uses learnt data representation to perform classification
Ensemble combine multiple learning algorithms to perform classification
Statistical model - HMM assume that the observed variables is dependent on the unknown variables and use this assumption to do the prediction
Statistical model - GMM allocate data points to multivariate normal for clustering

view on particular business and social issues [23]. Twitter sentiment analysis is applied on multiple domains, namely holiday season analysis [24], nuclear power generation [25], patient reactions to medicines [26], primary election [27], product brand [28], product sales [29], public health and epidemic outbreaks [30], stock market movements [31] and supreme court decision [32].

The myriad forms of Twitter sentiment analysis research have been performed using many different methods. Of the many methods used, recent sentiment analysis has been performed using machine learning [33], particularly supervised machine learning. Widely used machine learning algorithms which has been used for sentiment analysis includes rule-based algorithms [34], decision trees [35], support vector machines (SVM) [36], artificial neural networks [37], deep learning [38], ensembles [39], and statistical models such as the Hidden Markov model (HMM) [40] and the Gaussian mixture model (GMM) [41]. Table 1 shows a brief explanation for each supervised machine learning technique.

C. WORD EMBEDDINGS

Traditional text representation approaches like Bag-Of-Words (BOW), and Part Of Speech (POS) tagging have been surpassed by machine learning approaches, especially deep learning as deep learning has better classification accuracy [42]. Most deep learning models have focused on using word embeddings, which helps the learning algorithms to achieve better performance by grouping similar words [43]. Words are represented using word embedding based on their similarity in a corpus of relationships. There are two types of word embeddings, namely static [6], [44] and dynamic [45], [46] word embeddings.

1) STATIC WORD EMBEDDING

Static word embedding is considered static as these embeddings do not change with the context once it has been learned. Although static word embeddings are considered efficient, static word embeddings do not solve polysemy problems well, since the meaning of a polysemous word depends on its context [47].

Word2Vec (w2v) is an example of a static word embedding. It is implemented by using two-layer neural networks to create word representations. A large corpus of words is used in w2v, using either the Continuous Bag-Of-Words (CBOW) method which predicts the target word from surrounding words, or the Skip-Gram method which predict surrounding words from target word to generate word representations. Words that have similar context are located close to each other in the word representation vector space. However, w2v accounts only for local contexts [44].

Global Vectors for Word Representation (GloVe) is another static embedding method. A GloVe model is created as a log-bilinear regression model that has incorporated both the feature of global matrix factorization and the feature of local context window to create word representations. GloVe has improved on w2v by incorporating global word co-occurrence statistics of a corpus to generate word representations. In that way, GloVe can create word representation that captures meaning in the word vector space [6].

2) DYNAMIC WORD EMBEDDING

Text representation which is changed by context is known as contextualized embeddings, which are also known as dynamic word embeddings. These word embeddings are more effective in solving the polysemy problem as compared to static word embeddings [48].

One example of a dynamic embedding model is the Embeddings from Language Models (ELMo) [45]. BiLSTM is used in ELMo to create word representation. The entirety of the text is used when by ELMo when creating word representations, instead of considering every character in a text.

The BiLSTM used in ELMo allows it to distinguish word ambiguity well.

Bidirectional Encoder Representations from Transformers (BERT) [46] uses transformer to create word representations. Strong feature extraction and contextual expression ability are demonstrated in BERT. It fixes the overfitting problem by using masked language model (MLM). With MLM and next sentence prediction mechanism, it can generate good sentence representation.

In this article, we did not consider the usage of dynamic word embedding models, as we wanted to investigate how location information can provide additional context to the geo-tagged tweets. The ability of dynamic embedding models to provide dynamic context would have obscured the effect of concatenating location information to word embeddings.

D. DEEP LEARNING MODELS FOR SENTIMENT ANALYSIS

Popular deep learning models which have been used for sentiment analysis includes the CNN [49] and the RNN [50]. While it is difficult to come to a definitive conclusion as to which of the models perform better for sentiment analysis tasks, RNN has recorded a better performance in general [51].

1) CONVOLUTIONAL NEURAL NETWORK

CNN is composed of convolution layers with a feedforward neural network. Local features are extracted by the one-dimensional convolution layer as this layer has restricted the receptive fields of prior hidden layers [52]. Between neurons of adjacent layers, a local connectivity pattern is enforced to achieve spatially local correlation which is useful for text classification [53].

2) RECURRENT NEURAL NETWORK

Past input in an RNN is processed with the current input to result in a form of temporary memory which is effective at processing sequential information. A long sequence in RNN, however, is difficult to be processed due to short-term memory. Short-term memory is expressed as a difficulty to propagate information from earlier time steps to later ones. Short-term memory is caused by backpropagation through time (BPTT). The vanishing gradient problem is experienced by an RNN during backpropagation.

When a gradient value becomes extremely small, it does not contribute to much learning. Small gradient updates to layers in RNN will cause an RNN to stop learning. As a result of non-learning layers, RNN can 'forget' what it has seen in longer sequences, thus having a short-term memory. To overcome the shortcomings of RNN, researchers have developed more variants of RNN which includes the Bidirectional Recurrent Neural Network (BRNN) and Long Short-Term Memory Network (LSTM) [53].

3) BIDIRECTIONAL RECURRENT NEURAL NETWORK

BRNN [54] consists of two stacked RNNs. One layer processed the input in its original direction, while the other layer processed the input sequence in reversed direction. The hidden state of both RNN is then connected to form one output. Future input information is thus reachable from the current state. This increases the amount of input information available and solves the problem of RNN that the future input information cannot be reached from the current state.

4) LONG SHORT-TERM MEMORY NETWORK

Gates that can regulate the flow of information are found in LSTM [55]. Relevant information throughout the processing of the sequence is propagated through a LSTM. As a result, information from the earlier time steps can be propagated to later time steps, avoiding the vanishing gradient problem and reducing the short-term memory effects.

5) BIDIRECTIONAL LONG SHORT-TERM MEMORY NETWORK

Bidirectional Long Short-Term Memory Network (Bi-LSTM) combine the benefits of BRNN and LSTM. LSTM solve the vanishing gradient problem but only retain information from the past due to its input being one way. BiLSTM has both past (backward), and future (forward) input as it run inputs in two-way, one from future to past and one from past to future.

III. DATASET

A dataset with text and location categories is used to evaluate the performance of the discussed approach. The dataset is adapted from the public dataset hosted at the Carnegie Mellon University which contains 377616 English-only messages from 9475 geo-located microblog users approximately; within the United States; over one week. It contains five variables which are the anonymized user ID, time of tweet, latitude, longitude, and the tweet messages itself [56].

The latitude, longitude and text columns from this collection of geo-tagged tweets were used as the dataset. After the text is cleaned, rows with empty text or have a length of fewer than five characters is removed. The text data is pre-processed as follows [57]:

1) remove Unicode strings
2) convert URL (www, http…) to a token 'URL'
3) remove username and retweet
4) remove punctuation, number, and special characters
5) remove duplicated character that exceeds 3
6) remove additional white spaces

After the pre-processing steps has been completed, the dataset was labelled with a sentiment polarity label using the Valence Aware Dictionary and sEntiment Reasoner (VADER) [58]. VADER takes advantage of regulatory modelling approaches to develop an innovative sentiment analysis system that does not require training data and integrates a crowdsourcing methodology that enhances the lexical characteristics of candidates. VADER has exceeded the performance of individual human raters during sentiment analysis testing and performed more favourably across contexts for eleven state of the art sentiment benchmarks.

VADER includes a collection of ratings that include the positive score, neutral score, negative score, and the composite score. The compound score is a useful one-value metric calculating the sum of all standard lexicon ratings. Although VADER claimed to outperform human raters in predicting sentiment, another four other sentiment analysis libraries were tested to reduce the bias in this study.

The four sentiment analysis libraries which were tested were textblob [59], polyglot [60], IBM Watson natural language understanding [61] and senticnet [62]. Table 2 shows the findings after the evaluation process of each sentiment analysis library. Senticnet was excluded because it only works on a word-level rather than on a sentence-level. Polyglot and IBM Watson were excluded as these libraries required the input to be proper, i.e. these libraries cannot recognize short-form text, which is not suitable for twitter text.

TABLE 2. Libraries to predict sentiment polarity.

textblob Works on sentence but the polarity not accurate
vader Works on sentence
polyglot Works on sentence but language must be proper
ibm watson natural language understanding Input must exceed 15 characters and language must be proper
senticnet Only can work with word not sentence

After the evaluation process was completed, the conclusion was drawn that VADER is the most suitable library as it returned more accurate sentiment polarity score as compared to textblob. This conclusion was drawn after an assessment of five randomly sampled tweet was performed using VADER, textblob and human raters. Table 3 shows five rows of tweets with three sentiment outcomes. The human sentiment is author's opinion on the tweet, which is same as the result of VADER. Textblob has a different outcome, shown at row 4, which is bolded. From this assessment, VADER is a better choice for sentiment labelling as compared to textblob.

The tweets were then labelled according to the sentiment polarity score obtained using VADER, as shown in Algorithm 1. Two analytical datasets were created following the sentiment polarity labelling process. The first analytical dataset had multiclass (positive/negative/neutral) sentiment polarity labels, while the second analytical dataset contained binary (positive/negative) sentiment polarity labels.

TABLE 3. Comparison of human, vader, textblob sentiment on sampled tweet.

	text	human sentiment	vader sentiment	textblob sentiment
1	Why some niggaz put a front like they got shit an deep down not even dry shit dem nuh inna dem batty #Fakeassniggaz	negative	negative	negative
2	why is everybody and their momma in ATL this weekend	neutral	neutral	neutral
3	smirks well if you really don t need help okay But let me know if you DO need help I have a jack in the truck	positive	positive	positive
4	**#shoutout to for not having on all black today**	**neutral**	**neutral**	**negative**
5	Wow Have fun	positive	positive	positive

Algorithm 1 Categorization of Tweets to Sentiment

```
if score > 0 then
    sentiment = 2
else if score = 0 then
    sentiment = 1
else
    sentiment = 0
end if
```

The nearby location categories were acquired using the GeoNames Nearby web services [63] within a radius of 300 meters for each tweet. The tweets without nearby location categories were removed from the dataset, resulting in the multiclass dataset consisting of 17593 rows and the binary dataset consisting of 11521 rows of tweets. There is a total of 164 nearby location categories for the multiclass dataset (with positive, neutral, and negative labels) and 153 nearby location categories for the binary class dataset (with positive and negative labels).

The top three nearby location categories were buildings, churches, and schools with occurrences of 13342, 12730, and 11973 instances for the multiclass dataset and 9434, 8988, 8428 instances for the binary dataset respectively. The multi-class and binary-class geo-tweet dataset used in this study is collectively referred to as the Microblog dataset.

Table 4 summarize the details of Microblog dataset. Fig 2 shows that the Microblog dataset is a balanced dataset based on the class label histogram.

TABLE 4. Summary of details of the Microblog dataset.

	Multiclass	Binary
Tweets number	17593	11521
Positive	6612	6612
Neutral	6072	-
Negative	4909	4909
Categories count	164	153
Buildings	13342	9434
Church	12730	8988
School	11973	8428

IV. METHODOLOGY FOR TWITTER TEXT SENTIMENT CLASSIFICATION

In this section, the text classification process using deep learning is explained. The methodology used for performing the baseline twitter text sentiment classification is described, followed by a description of the deep learning architecture used to perform the experiment.

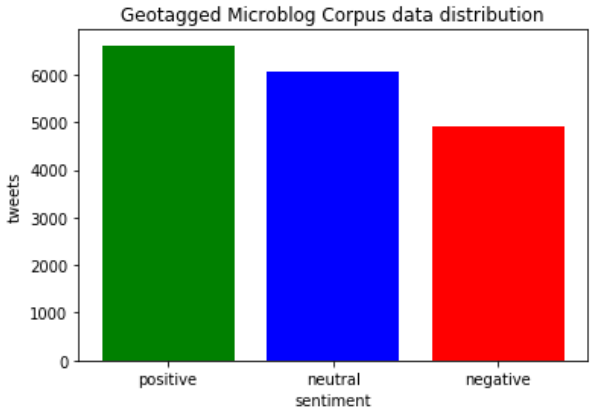


FIGURE 2. Distribution of data in the Microblog dataset.

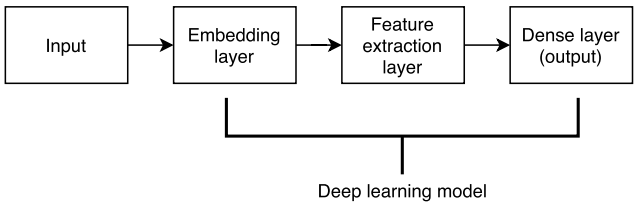


FIGURE 3. Text classification process.

The input twitter text must be first converted from a text to a numeric representation. The representation is then used as an input for a deep learning model to continue the classification process. Fig 3 show the text classification process using deep learning. The first layer in the deep learning model is the embedding layer, followed by the feature extraction layer with a final dense (output) layer.

A. EMBEDDING LAYER

The input twitter text will be integer encoded by assigning each word to a unique integer number (by order of appearance in the dataset). The integer encoded representation is subsequently padded into an uniform length. This is then sent through the embedding layer, which acts as a lookup table to generate a weight matrix based on the integer encoded input.

The size of the vocabulary, output dimension and embedding matrix of pre-trained GloVe model are essential to ensure the creation of an accurate lookup table [64]. The size of the vocabulary is the occurrences of unique words in the dataset. The output dimension is the size of the vector space in which words will be embedded. Lastly, the dimension of the embedding matrix is dependent on the pre-trained GloVe model.

The embedding layer creates embedding vectors from the input twitter text. It compresses the input feature space into a smaller space by finding an optimal mapping of each of the unique words to a vector of real numbers [65]. The embedding matrix keeps the vector size smaller, which promote efficient computation. The product obtained from this layer forms the feature which are then passed to the feature extraction layer.

As CNN and BiLSTM are using different feature extraction techniques, the feature extraction layer for CNN and BiLSTM is discussed in Section C and Section D respectively.

B. OUTPUT LAYER

The classification process is enabled by the dense layer, which is also known as a fully-connected layer. Activation functions are used to determine the output of the deep neural network. The sigmoid activation function is suitable for binary classification as the output is in zero and one, while the softmax activation function is suitable for multiclass classification as the output is zero, one and two.

C. CNN

The CNN used in this study consists of an embedding layer, three convolution layers, a pooling layer, a flatten layer, and finally an output layer, as shown in Fig 4.

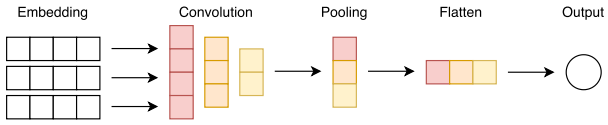


FIGURE 4. CNN structure used in this research.

Features from the twitter text are extracted by the convolution layer. The convolution layer contains a kernel or filter that is a weight that slides through the input sentence matrix. The output of this process is a matrix called the feature map, which is the dot product of the input matrix and weight. The size of the feature map is controlled by several filters, padding, and stride, which is unit per slide.

The amount of data from the feature map is reduced by the pooling layer. The pooling layer abstracts information to improve generalization. Max-pooling has performed better than average and minimum poling [66]. Max-pooling is a pooling operation that calculates the largest value in each window of the feature map.

A single long feature vector is created as input to the dense layer. This feature vector is a result of the flatten layer, which has converted the three-dimensional feature from the pooling layer into a one-dimensional feature as shown in Fig 4.

D. BiLSTM

The BiLSTM used in this study consists of an embedding layer, with one LSTM in a forward direction and one LSTM in a backward direction and lastly an output layer, as shown in Fig 5. One LSTM accesses past information in the forward direction while another LSTM accesses future information in the reverse direction. There is no need to use the Flatten layer as the LSTM output is two-dimensional.

The component of the LSTM used in the BiLSTM is shown in Fig 6. In the forward LSTM layer, the information flow in one direction as in Fig 6. For the backward LSTM layer, the information flow is in the another direction.

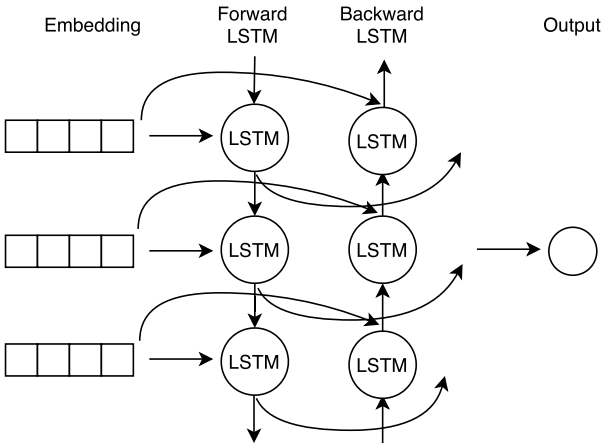


FIGURE 5. BiLSTM structure used in this research.

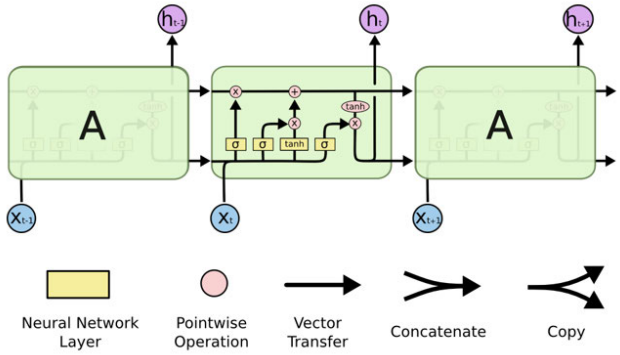


FIGURE 6. Structure of LSTM [67].

The current input and the previous output are channelled through the Forget gate, which contains a sigmoid layer to decide what information is to be discarded. The output is an either zero or one, where one is to keep, and zero is to forget. Subsequently, a choice is made on the information that will be stored in the cell state.

The sequence goes to Input gate which contains a sigmoid layer decide which value to keep from tanh layer by multiplying the tanh output with sigmoid output. The previous cell state gets pointwise multiplied by the Forget gate output. The new output then gets pointwise addition by the Input gate output to get a new cell state.

The next hidden state is decided by the Output gate which contains a sigmoid layer. The hidden state contains information about previous inputs, and it is also used for prediction. The previous hidden state and current input are passed to a sigmoid function. The new cell state obtained earlier is also passed to a tahn function. Both outputs are multiplied to get a new hidden state. The new cell state and new hidden state is then carried over to the next LSTM unit.

V. PROPOSED METHOD TO INCORPORATE LOCATION INFORMATION AS A FEATURE FOR TWITTER TEXT SENTIMENT CLASSIFICATION

In this section, the method used to incorporate location information as a feature for twitter text sentiment classification is presented. Two different approaches were used to represent location information in this study, namely frequency-count encoding and vectorized ego-network.

A. LOCATION VECTORIZATION

The purpose of this approach is to represent location categories through frequency-count encoding. As an example, there are 164 nearby location categories in the multiclass Microblog dataset. Frequency-count encoding is applied to the location categories instead of one-hot encoding, resulting in a 164-dimension matrix. Our previous study has shown that frequency-count encoding has resulted in higher accuracy for twitter sentiment classification as compared to one-hot encoding [68]. The text representation matrix is concatenated with the nearby location categories matrix as the input to the deep learning model.

Concatenation of different feature vectors is an approach which has been used successfully for twitter sentiment analysis, with previous efforts focusing on the concatenation of word embeddings and n-grams features and word sentiment polarity score features [69]. In this study, we used two approaches for concatenating twitter text with the nearby location categories. Concatenation can be performed either before the embedding layer, or concatenation occurs after both inputs goes through the embedding layer.

1) TEXT CONCATENATED WITH LOCATION AS INPUT

To increase the location context of tweets, the nearby location categories for each geo-coded tweet is concatenated with the twitter text representation, as the input to the embedding layer. As shown in Fig 7, the concatenated input passes through an embedding layer with pre-trained word embedding model weight before being used as an input for a deep learning model.

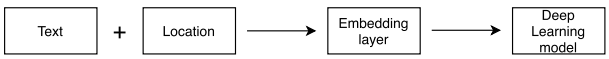


FIGURE 7. Approach of text concatenate with location as input.

2) TEXT AND LOCATION AS SEPARATE INPUT

Rather than concatenate nearby location categories with twitter text immediately, the twitter text representation and nearby location for each geo-coded tweet are treated as separate inputs. As shown in Fig 8, the twitter text is passed through an embedding layer with pre-trained word embedding model weight while the nearby location information is passed through an embedding layer with no word embedding model. The output generated is then concatenated through a merged layer before being used as an input for a deep learning model.

B. LOCATION AS A VECTORIZED EGO NETWORK

Although there is an improvement in the sentiment classification accuracy via concatenating text with nearby location categories, it is an inefficient approach as the size of the data

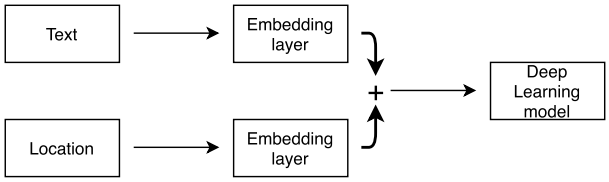


FIGURE 8. Approach of text and location as input.

increases. As the input matrix becomes sparse; the resultant feature dimension also increases, which leads to increased training time. To solve this problem, dimension reduction is necessary.

As the tweets and location categories has hierarchical pattern, a hierarchy based dimension reduction technique is applied to it. Inspired from social network analysis, the individual tweet is represented as an ego network, which combines the perspective of network analysis with the aproaches of mainstream social science [70]. As an example, an ego network of a tweet that has a nearby restaurant, school and a library is shown in Fig 9. The circle is called a node, and the edge linked between node is called a tie. An ego (Tweet) is connected by three nodes (location categories).

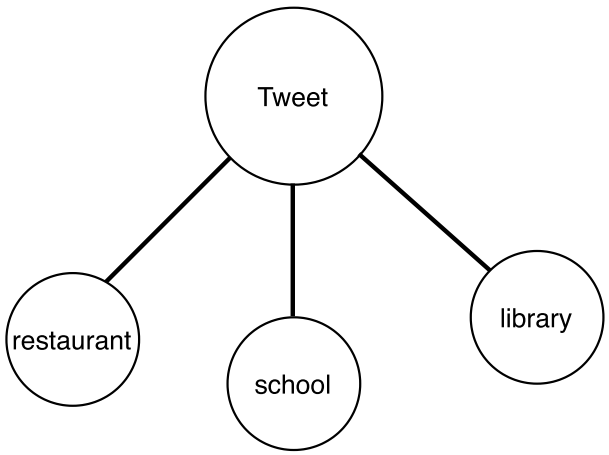


FIGURE 9. Tweets and location in graph.

Three measurements from ego networks were adopted for the purpose of dimension reduction. The three measurements used in our study were network density, closeness centrality, and degree centrality [71]. In our study, the 164 nearby location categories from the multi-class Microblog dataset can be reduced to three measurements, which make it consistent no matter the number of nearby location categories. Equations 1 to 3 shows the calculation for ego network related feature vectors.

$$\textit{network density} = \frac{\textit{number of ties}}{\textit{number of node} * ((\textit{number of node} - 1)/2)} \quad (1)$$

$$\textit{closeness centrality} = \frac{1}{\textit{sum of distance to other nodes}} \quad (2)$$

$$\textit{degree centrality} = \textit{number of ties that touch a node} \quad (3)$$

1) TEXT CONCATENATED WITH EGO NETWORK FEATURES AS INPUT

The method used to concatenate text with ego network features is the same as the method described in Section V.A.1, with the difference that the nearby location feature is replaced by the three ego-network related measurements as shown in Fig10.

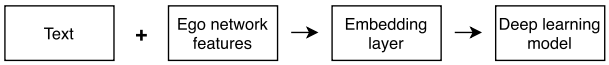


FIGURE 10. Approach of text concatenated with ego network features as input.

2) TEXT AND EGO NETWORK FEATURES AS SEPARATE INPUT

The method used to concatenate text with ego network features as separate features is the same as the method described in V.A.2 with the difference that the nearby location feature is replaced by the three ego-network related measurements as shown in Figure 11.

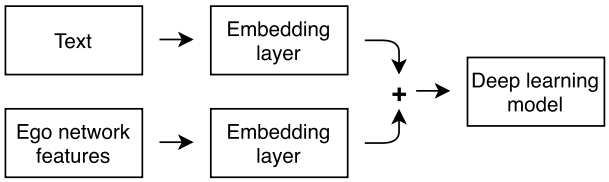


FIGURE 11. Approach of using both of text and ego network features as input.

VI. RESULTS

In this section, the result of empirical experiments is presented. These experiments were performed to evaluate the outcome of our method which concatenates location information with text as a feature to be used in a twitter sentiment classification task.

Our preliminary study [68] which had incorporated location information for sentiment classification tasks have shown an improvement in the classification accuracy. The frequency count of nearby categories approaches led to higher accuracy as compared to one-hot encoding for vectorized nearby location categories. The best accuracy result is achieved using a pre-trained GloVe model trained on 27B Twitter data with 200 vector dimension on the embedding layer.

Each experiment in this study was conducted using Python 3.6 in Linux Mint 19.1 with a seed value of 7. The dataset is loaded using pandas [72]. Data processing is done using NumPy [73] and Keras [74] with a Tensorflow backend [75]. The dataset is split into training and test set with the train-test split ratio of 7:3 [76]. The final accuracy is the average

TABLE 5. Experiment result of binary classification on CNN.

	Accuracy			Loss			Training time
Input	Max	Min	Mean	Max	Min	Mean	(seconds)
Text	0.8124	0.8061	0.8108	1.6597	0.6317	1.3623	41.08438
Text concatenated with location categories	0.8547	0.8258	0.8511	1.4708	0.7871	1.2503	361.2392
Text and location categories as separate input	0.8518	0.8311	0.8447	1.4555	0.6843	1.2104	416.7792
Text concatenated with ego network measurements	0.8471	0.8366	0.8449	1.387	0.6989	1.1528	109.8327
Text and ego network measurements as separate input	0.8447	0.8248	0.8398	1.6431	0.8281	1.4249	162.5174

TABLE 6. Experiment result of binary classification on BiLSTM.

	Accuracy			Loss			Training time
Input	Max	Min	Mean	Max	Min	Mean	(seconds)
Text	0.8862	0.8642	0.8769	0.6963	0.3391	0.5599	345.1042
Text concatenated with nearby location categories	0.8828	0.8487	0.8627	1.4536	0.3287	0.9365	1109.945
Text and nearby location categories as separate input	0.8854	0.8568	0.8625	1.2299	0.3524	0.9637	1321.12
Text concatenated with ego network measurements	0.8776	0.8434	0.8564	1.5074	0.3499	1.0296	260.7753
Text and ego network measurements as separate input	0.8791	0.8431	0.8552	1.5132	0.3794	1.0394	373.6193

accuracy over 20 runs. The maximum length for text is 30, while the location length depends on the experimental setting. Experiment with text-only input is considered the baseline experiment; while validation of our approach is achieved upon the achievement of an improvement in terms of accuracy for subsequent twitter sentiment classification tasks performed using the concatenated location-text features. Other measures used to compare the performance of the concatenated features are model training time and loss.

The architecture of the deep learning model used in this study is described as follows: The CNN starts with three convolution layers; each has 100 filters with ReLU activation function and a kernel size of 5, 4, 3, respectively. Next is the max-pooling layer with a pool size of 2. There is a flatten layer followed the output layer with the activation function of sigmoid for binary classification and softmax for multiclass classification. The BiLSTM has LSTM layer with 100 neurons and 0.5 dropout rate. The output layer is the same as CNN settings.

Adam is used to optimize the deep learning model. Loss is calculated using binary cross-entropy for binary classification and sparse categorical cross-entropy for multiclass classification. Validation split of 7:3 and 20 epochs was used in training the model. Early stopping is set to stop training upon reaching three epochs without improvement to prevent overfitting.

A. BINARY CLASSIFICATION USING CNN

Table 5 shows that all input method exceeds the baseline accuracy for binary classification performed using CNN. The highest accuracy achieved was 85.11% using text concatenated with nearby location categories as input. The accuracy is improved by 5.1% as compared to baseline accuracy, however, it takes 6 minutes to train. The overall best input method is text concatenated with the ego network measurements which have the second-highest accuracy of 87.57% and the lowest loss of 1.1528, and it takes only 1.8 minutes to train.

B. BINARY CLASSIFICATION USING BiLSTM

Table 6 shows that adding location categories to text actually reduced the classification accuracy, which suggest that text only features is most useful for achieving high accuracy for binary classification of twitter text performed using BiLSTM. Although the classification accuracy did not improve, the time to train the model is reduced by using text concatenate with ego network measurements as compared to using text concatenated with nearby location features.

The sentiment classification experiment performed using BiLSTM with a binary label resulted in a mean accuracy of 87.69% and a mean loss of 0.5599. When text is concatenated with nearby location categories as a feature vector, the mean accuracy is reduced to 86.27%, and the mean loss has increased to 0.9365.

C. MULTICLASS CLASSIFICATION USING CNN

Table 7 shows that the highest accuracy for multiclass classification on CNN is 84.26% using text concatenated with nearby location categories as input. This score improves by 13.8% compared to the baseline accuracy, but it takes 9.3 minutes to train. The overall best input method is text concatenated with ego network measurements which have an accuracy of 83.99%, and it takes only 2.8 minutes to train.

D. MULTICLASS CLASSIFICATION USING BiLSTM

Table 8 shows that the highest accuracy for multiclass classification on BiLSTM is 87.77% using text and nearby location categories as separate input. This score improves by 4.1% compared to the baseline accuracy, but it takes 37.8 minutes to train. The overall best input method is text concatenate with ego network measurements which have an accuracy of 87.57%, and it takes only 7.3 minutes to train.

E. SUMMARY

From the results of the experiments, BiLSTM has better classification accuracy than CNN. On both the CNN models,

TABLE 7. Experiment result of multiclass classification on CNN.

	Accuracy			Loss			Training time
Input	Max	Min	Mean	Max	Min	Mean	(seconds)
Text	0.7539	0.7202	0.7404	2.5637	1.0903	2.2236	59.77406
Text concatenated with location categories	0.8452	0.8246	0.8426	1.5593	0.8848	1.3451	560.4678
Text and location categories as separate input	0.8393	0.8121	0.8365	1.6761	0.9229	1.4638	638.3973
Text concatenated with ego network measurements	0.8427	0.8179	0.8399	1.5998	0.8849	1.3976	168.0574
Text and ego network measurements as separate input	0.8352	0.8026	0.8302	1.7421	0.9479	1.5578	254.9599

TABLE 8. Experiment result of multiclass classification on BiLSTM.

	Accuracy			Loss			Training time
Input	Max	Min	Mean	Max	Min	Mean	(seconds)
Text	0.8537	0.8264	0.8435	0.8402	0.4693	0.6844	578.2734
Text concatenated with location categories	0.8844	0.8587	0.8727	1.2954	0.3831	0.9825	1983.731
Text and location categories as separate input	0.8907	0.8615	0.8777	1.1943	0.4031	0.9177	2271.619
Text concatenated with ego network measurements	0.8897	0.8638	0.8757	1.2777	0.3695	0.9795	435.3295
Text and ego network measurements as separate input	0.8825	0.8537	0.8729	1.3633	0.4131	0.992	565.137

adding location has increased the accuracy and reduced the loss, which indicates that the addition of location information has improved the CNN model performance. The increase of accuracy and decrease of loss can be interpreted as the increased frequency of correctly classified labels and the decrease of the confidence level of the model respectively.

However, the behaviour of BiLSTM is different for the binary and the multiclass dataset. For the binary BiLSTM experiments, the classification accuracy is high (highest among all experiments which is 87.69%) and adding location information to text has actually decreased the classification accuracy and increased the loss.

Conversely, the classification accuracy and the loss of multiclass BiLSTM experiments has increased when adding location information. The increased of accuracy and also loss can be interpreted as the increased frequency of correctly classified labels and subsequently the decrease of the confidence level of the model respectively.

The possible reason of the experiments result could be that the BiLSTM have more suitable feature extraction method in taking sequence data as compared to the CNN model. Thus, the classification capability is better. More detailed discussion of the experimental results is presented in next section.

TABLE 9. Comparison of SHAP values on CNN binary classification.

text only	4, 2, 5, 6, 1
concatenated	5, 1, 2, 0, 4

VII. DISCUSSION

In this section, a discussion of the impact of concatenating nearby location information to text features is discussed. In this study, we attempted to investigate the effect of adding location as context for twitter text sentiment classification via investigation of feature importance.

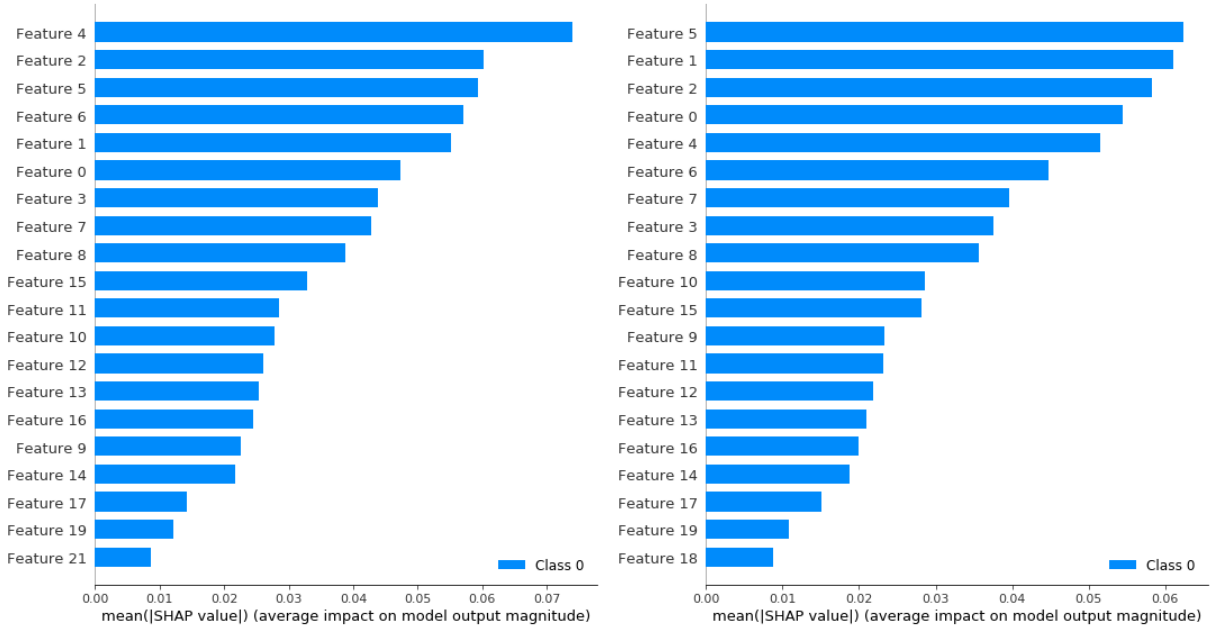
It is beneficial to know which feature the deep learning model deems to be necessary. A comparison of before and after adding the location feature to the text should give some insights on the feature weightage. To understand the deep learning model, SHapley Additive exPlanations (SHAP) is used to visualize the feature. It connects optimal credit allocation with local explanations using the classical Shapley values from game theory and their related extensions [77]. The higher the mean SHAP values of a feature, the higher the contribution of that feature to the model.

Since only the category of binary classification using BiLSTM shows different behaviour, SHAP is applied to the binary classification model to see the feature weight. The input method of text-only and text concatenated with ego network measurements are compared for simple visualisation as the former have 30 features while the latter have 33 features. The 0th feature to 30th feature are text features, while the 31st feature to 33rd feature are location features. The first 100 data from the training set is fitted to the SHAP explainer and is then used on the first 100 data from the testing set to generate the mean SHAP value.

In the binary classification CNN model, Fig 12 shows the mean SHAP values for the model that uses text as input that is having an accuracy of 80.21% while the model that used text concatenated with ego network measurements as input having an accuracy of 83.10%.

A comparison of the top mean SHAP value that has exceeded 0.05 of the CNN model (taken from Fig 12) is shown in Table 9. It appears that CNN did not take nearby location categories as an essential feature as the result generated by SHAP only shows features associated with text only features. The concatenation of nearby location categories as a feature has resulted in a change in the feature order, which corresponds to an increase in the classification accuracy. The unique features on the text only model is the 6th feature while on the concatenated model the 0th feature is unique, which can be interpreted as model that put more weightage on the 0th feature will result in an increase in accuracy.

In the binary classification BiLSTM model, Fig 13 shows the model that uses text as input that has an accuracy



(a) Text only - 80.21%

(b) Text concatenate ego network measurements - 83.10%

FIGURE 12. SHAP on binary classification CNN.

TABLE 10. Comparison of SHAP values on BiLSTM binary classification.

text only	0, 1, 2, 4, 5, 7
concatenated	0, 1, 5, 2, 6

of 87.86% while the model that uses text concatenate with ego network measurements as input have an accuracy of 86.73%.

A comparison of the top SHAP features with mean SHAP values exceeding 0.05 of the BiLSTM model (taken from Fig 13) is shown in Table 10. It appears that BiLSTM also did not take location categories as an essential feature as the result generated by SHAP only shown features until the 19th feature. Adding nearby location feature also swap the priority of feature that increase the classification accuracy. The findings in the CNN model shown consistency which put more weightage on the 0th feature and less weightage on the 6th feature, which increase the accuracy.

From the SHAP results of CNN and BiLSTM, feature 0th, 1st, 2nd, 4th, 5th have high weightage among all the features. Input text has 30 features; SHAP also shows that all model only consider feature within it. Adding nearby location feature with text won't let the model take it as the feature to calculate prediction but will change the order of feature prioritization which increase the classification accuracy. The improvements is only applied when the model is not in optimal performance.

Pearson's correlation was performed among the three ego network measurements to see the relationship between them. Table 11 shows the results of Pearson's correlation. The ego

TABLE 11. Pearson's correlation of ego network measurements.

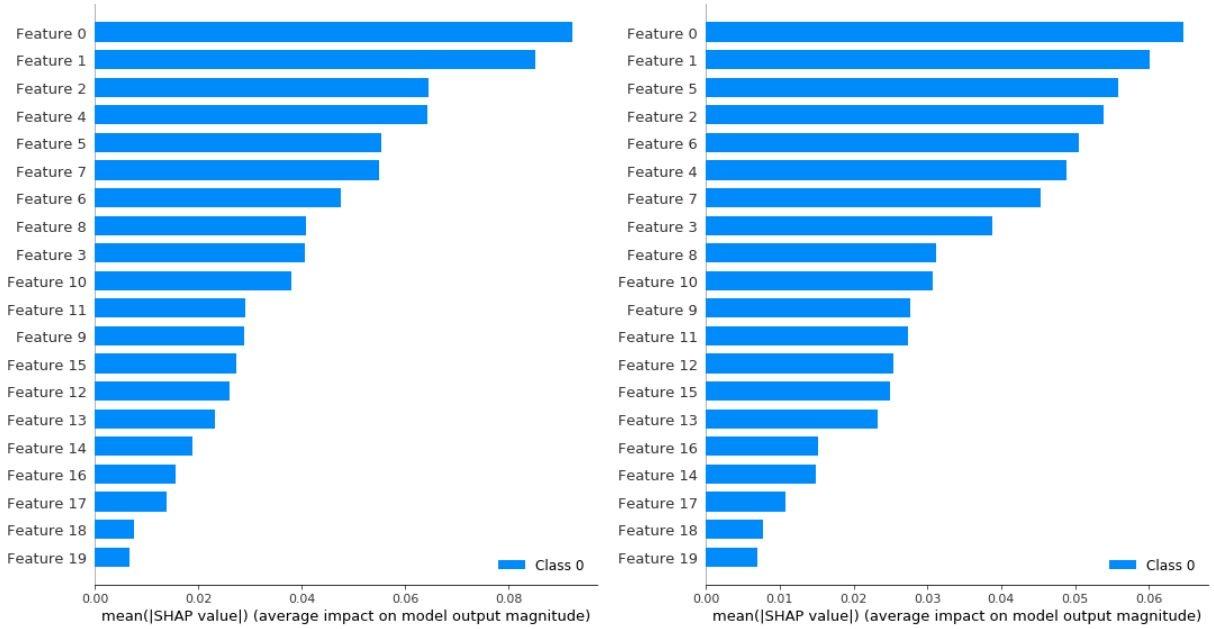
	ego_density	ego_closeness	ego_degree
ego_density	-	4.39e-15	-7.32e-15
ego_closeness	4.39e-15	-	-8.24e-01
ego_degree	-7.32e-15	-8.24e-01	-

TABLE 12. Sentiment of selected location categories occurrences in the dataset.

	positive	neutral	negative
Park	2041	1801	1987
Railroad station	20	14	24
Hospital	540	405	475

density has very low positive linear correlation with the ego closeness and vice versa. The ego density also has very low linear negative correlation to the ego degree and vice versa. The ego closeness has a high linear negative correlation to the ego degree and vice versa. From the findings, the three features are not similar and are viable to be retained as features.

Table 12 shows the selected three location categories occurrences and the sentiment associated. From the reviewed studies, park and medical centres have high sentiment values and transportation hub have low sentiment values. The dataset used in this experiment showed this statement is true by the occurrences of the location categories. However, there is no one category that can be concluded to be associated with positive or negative sentiment as the ratio of each type of sentiment is nearly equal.



(a) Text only - accuracy of 87.86%

(b) Text concatenate ego network measurements - 86.73%

FIGURE 13. SHAP on binary classification BiLSTM.

VIII. CONCLUSION

Given a geo-tagged tweet that has latitude and longitude with it, nearby location categories can be used as a feature. To use nearby location categories with text, it can be concatenated with the text as an input to the predictive model. Just by concatenating location categories with text as input can improve the classification accuracy, however, this approach is not viable in the long term. The input is a sparse vector that has many columns.

As the number of nearby location categories increases, the training time takes longer. To maintain the performance, an approach of taking tweet and location categories as ego network is suggested. The input vector goes through dimension reduction by using three measurements from ego network analysis; as such no matter how much the location categories increase, the result is still three columns.

The experiments were performed on four categories with two on binary classification and two on multiclass classification. Among the four categories, only binary classification on BiLSTM shows no improvement when concatenating nearby location categories to text. The other three categories show a consistency whereby the concatenation of nearby location categories and ego network measurements of all input methods is better than using text only as input. The best input method is text concatenated with ego network measurements as it yields good accuracy, and has shorter training time. The highest accuracy method, which concatenates the text with nearby location categories, is taking a long time.

When compared to concatenating text with ego network measures, the time taken by models which concatenates nearby location categories to text, has resulted in training time increasing from between 228% to 422%. When the amount of data is increasing, the number of location categories is likely to increase, which means the training time using location categories is becoming longer. If using text-only already provide an acceptable result, there is no need to add location categories to it, unless shorter training time is needed. While the result still can be better, with the training time into consideration, the most suitable method is text concatenate with ego network measurements as input and feed into the deep learning model.

A. FUTURE WORK

From the literature review and implementation, we have come out with few future research directions that are worth considering:

1) More metrices to characterize urban typologies such as street based metrics [78] can be investigated.
2) Use dynamic embedding on the text and make use of the Transformer model.
3) Try other operations [79] like addition or multiplication rather than concatenate in the merge layer of neural network.
4) Treat the tweet, and nearby location categories as a graph then use node embedding approaches like node2vec and random walk to generate the embedding.
5) Include more properties to location data such as distance between a tweet to a location.
6) Use graph neural network to perform classification.

REFERENCES

[1] B. Pang, L. Lillian, and V. Shivakumar, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proc. ACL Conf. Empirical Methods Natural Lang. Process.*, vol. 10, Jul. 2002, pp. 79–86.

[2] H. J. Do and H.-J. Choi, "Sentiment analysis of real-life situations using location, people and time as contextual features," in *Proc. Int. Conf. Big Data Smart Comput. (BIGCOMP)*, Feb. 2015, pp. 39–42.

[3] E. Cambria, S. Poria, A. Gelbukh, and M. Thelwall, "Sentiment analysis is a big suitcase," *IEEE Intell. Syst.*, vol. 32, no. 6, pp. 74–80, Nov. 2017.

[4] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: http://arxiv.org/abs/1408.5882

[5] A. Varghese, G. Agyeman-Badu, and M. Cawley, "Deep learning in automated text classification: A case study using toxicological abstracts," *Environ. Syst. Decisions*, pp. 1–15, Feb. 2020.

[6] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[7] Z. Sadeghi, J. L. McClelland, and P. Hoffman, "You shall know an object by the company it keeps: An investigation of semantic representations derived from object co-occurrence in visual scenes," *Neuropsychologia*, vol. 76, pp. 52–61, Sep. 2015.

[8] E. Hauthal and D. Burghardt, "Mapping space-related emotions out of user-generated photo metadata considering grammatical issues," *Cartograph. J.*, vol. 53, no. 1, pp. 78–90, Jan. 2016.

[9] K. Z. Bertrand, M. Bialik, K. Virdee, A. Gros, and Y. Bar-Yam, "Sentiment in new york city: A high resolution spatial and temporal view," 2013, *arXiv:1308.5010*. [Online]. Available: http://arxiv.org/abs/1308.5010

[10] H. Roberts, J. Sadler, and L. Chapman, "The value of Twitter data for determining the emotional responses of people to urban green spaces: A case study and critical evaluation," *Urban Stud.*, vol. 56, no. 4, pp. 818–835, Mar. 2019.

[11] J. B. Hollander and H. Renski, "Measuring urban attitudes embedded in microblogging data: Shrinking versus growing cities," *Town Planning Rev.*, vol. 88, no. 4, pp. 465–490, Jul. 2017.

[12] W. B. D. Oliveira, L. B. Dorini, R. Minetto, and T. H. Silva, "OutdoorSent: Sentiment analysis of urban outdoor images by using semantic and deep features," *ACM Trans. Inf. Syst.*, vol. 38, no. 3, pp. 1–28, Jun. 2020.

[13] F. Ali, D. Kwak, P. Khan, S. M. R. Islam, K. H. Kim, and K. S. Kwak, "Fuzzy ontology-based sentiment analysis of transportation and city feature reviews for safe traveling," *Transp. Res. C, Emerg. Technol.*, vol. 77, pp. 33–48, Apr. 2017.

[14] T. Hu, B. She, L. Duan, H. Yue, and J. Clunis, "A systematic spatial and temporal sentiment analysis on geo-tweets," *IEEE Access*, vol. 8, pp. 8658–8667, 2020.

[15] A. Jain and M. Jain, "Location based Twitter opinion mining using common-sense information," *Global J. Enterprise Inf. Syst.*, vol. 9, no. 2, p. 28, Jun. 2017.

[16] P. Singh, R. S. Sawhney, and K. S. Kahlon, "Sentiment analysis of demonetization of 500 & 1000 rupee banknotes by indian government," *ICT Express*, vol. 4, no. 3, pp. 124–129, Sep. 2018.

[17] B. Hu and M. Ester, "Social topic modeling for Point-of-Interest recommendation in location-based social networks," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2014, pp. 845–850.

[18] L. Hu, A. Sun, and Y. Liu, "Your neighbors affect your ratings: On geographical neighborhood influence to rating prediction," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2014, pp. 345–354.

[19] B. Liu, "Sentiment analysis and subjectivity," *Handbook Natural Lang. Process.*, vol. 2, no. 2010, pp. 627–666, 2010.

[20] M. M. Miroáczuk and J. Protasiewicz, "A recent overview of the state-of-the-art elements of text classification," *Expert Syst. Appl.*, vol. 106, pp. 36–54, Sep. 2018.

[21] M. V. Mäntylä, D. Graziotin, and M. Kuutila, "The evolution of sentiment analysis—A review of research topics, venues, and top cited papers," *Comput. Sci. Rev.*, vol. 27, pp. 16–32, May 2018.

[22] W. Ahmed, "Using twitter as a data source: An overview of social media research tools (updated for 2017)," *Impact Social Sci. Blog*, May 2017.

[23] D. Zimbra, A. Abbasi, D. Zeng, and H. Chen, "The state-of-the-art in Twitter sentiment analysis: A review and benchmark evaluation," *ACM Trans. Manage. Inf. Syst.*, vol. 9, no. 2, pp. 1–29, 2018.

[24] W. Hu, "Real-time Twitter sentiment toward thanksgiving and christmas holidays," *Social Netw.*, vol. 2, no. 2, pp. 77–86, May 2013.

[25] D. Kim and J.-W. Kim, "Public opinion mining on social media: A case study of Twitter opinion on nuclear power," *Adv. Sci. Technol. Lett.*, vol. 51, pp. 224–228, Jun. 2014.

[26] A. Abbasi, D. Adjeroh, M. Dredze, M. J. Paul, F. M. Zahedi, H. Zhao, N. Walia, H. Jain, P. Sanvanson, R. Shaker, M. D. Huesch, R. Beal, W. Zheng, M. Abate, and A. Ross, "Social media analytics for smart health," *IEEE Intell. Syst.*, vol. 29, no. 2, pp. 60–80, Mar./Apr. 2014.

[27] Y. Mejova, P. Srinivasan, and B. Boynton, "Gop primary season on Twitter: Popular political sentiment in social media," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 517–526.

[28] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury, "Twitter power: Tweets as electronic word of mouth," *J. Amer. Soc. for Inf. Sci. Technol.*, vol. 60, no. 11, pp. 2169–2188, Nov. 2009.

[29] H. Rui, Y. Liu, and A. Whinston, "Whose and what chatter matters? The effect of tweets on movie sales," *Decis. Support Syst.*, vol. 55, no. 4, pp. 863–870, 2013.

[30] X. Ji, S. A. Chun, and J. Geller, "Monitoring public health concerns using Twitter sentiment classifications," in *Proc. IEEE Int. Conf. Healthcare Informat.*, Sep. 2013, pp. 335–344.

[31] J. Smailović, M. Grčar, N. Lavrač, and M. Žnidaršič, "Predictive sentiment analysis of tweets: A stock market application," in *Proc. Int. Workshop Human-Computer Interact. Knowl. Discovery Complex, Unstruct., Big Data*, 2013, pp. 77–88.

[32] T. S. Clark, J. K. Staton, E. Agchtein, and Y. Wang, "Revealed public opinion on Twitter: The supreme court of the united states same-sex marriage decisions," Emory University, Atlanta, GA, USA, Tech. Rep., 2014.

[33] L. Yue, W. Chen, X. Li, W. Zuo, and M. Yin, "A survey of sentiment analysis in social media," *Knowl. Inf. Syst.*, vol. 15, pp. 1–47, Oct. 2018.

[34] M. Z. Asghar, A. Khan, S. Ahmad, M. Qasim, and I. A. Khan, "Lexicon-enhanced sentiment analysis framework using rule-based classification scheme," *PLoS ONE*, vol. 12, no. 2, Feb. 2017, Art. no. e0171649.

[35] V. N. Phu, V. T. N. Tran, V. T. N. Chau, N. D. Dat, and K. L. D. Duy, "A decision tree using ID3 algorithm for english semantic analysis," *Int. J. Speech Technol.*, vol. 20, no. 3, pp. 593–613, Sep. 2017.

[36] Y. Liu, J.-W. Bi, and Z.-P. Fan, "A method for multi-class sentiment classification based on an improved one-vs-one (OVO) strategy and the support vector machine (SVM) algorithm," *Inf. Sci.*, vols. 394–395, pp. 38–52, Jul. 2017.

[37] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1422–1432.

[38] D. Tang, F. Wei, B. Qin, T. Liu, and M. Zhou, "Coooolll: A deep learning system for Twitter sentiment classification," in *Proc. 8th Int. Workshop Semantic Eval. (SemEval)*, 2014, pp. 208–212.

[39] O. Araque, I. Corcuera-Platas, J. F. Sánchez-Rada, and C. A. Iglesias, "Enhancing deep learning sentiment analysis with ensemble techniques in social applications," *Expert Syst. Appl.*, vol. 77, pp. 236–246, Jul. 2017.

[40] M. Kang, J. Ahn, and K. Lee, "Opinion mining using ensemble text hidden Markov models for text classification," *Expert Syst. Appl.*, vol. 94, pp. 218–227, Mar. 2018.

[41] M. Abdel Fattah, "New term weighting schemes with combination of multiple classifiers for sentiment analysis," *Neurocomputing*, vol. 167, pp. 434–442, Nov. 2015.

[42] Y. Zhang, Z. Zhang, D. Miao, and J. Wang, "Three-way enhanced convolutional neural networks for sentence-level sentiment classification," *Inf. Sci.*, vol. 477, pp. 55–64, Mar. 2019.

[43] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[44] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781

[45] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018, *arXiv:1802.05365*. [Online]. Available: http://arxiv.org/abs/1802.05365

[46] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805

[47] Y. Wang, Y. Hou, W. Che, and T. Liu, "From static to dynamic word representations: A survey," *Int. J. Mach. Learn. Cybern.*, vol. 12, pp. 1–20, Feb. 2020.

[48] R. Bamler and S. Mandt, "Dynamic word embeddings," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 380–389.

[49] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, *arXiv:1404.2188*. [Online]. Available: http://arxiv.org/abs/1404.2188
[50] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.
[51] W. Yin, K. Kann, M. Yu, and H. Schätze, "Comparative study of CNN and RNN for natural language processing," 2017, *arXiv:1702.01923*. [Online]. Available: http://arxiv.org/abs/1702.01923
[52] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.
[53] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 8, no. 4, p. e1253, 2018.
[54] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.
[55] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[56] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2010, pp. 1277–1287.
[57] A. Kulkarni and A. Shivananda, *Natural Language Processing Recipes: Unlocking Text Data With Machine Learning and Deep Learning Using Python*. New York, NY, USA: Apress, Jan. 2019.
[58] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Proc. 8th Int. AAAI Conf. Weblogs Social Media*, 2014, pp. 1–5.
[59] S. Loria. *Simple, Pythonic Text Processing. Sentiment Analysis, Part-of-Speech Tagging, Noun Phrase Parsing, and More*. Accessed: Aug. 4, 2019. [Online]. Available: https://pypi.org/project/textblob/
[60] R. Al-Rfou. *Polyglot*. Accessed: Sep. 12, 2019. [Online]. Available: https://github.com/aboSamoor/polyglot
[61] *Watson Natural Language Understanding*. Accessed: Sep. 13, 2019. [Online]. Available: https://www.ibm.com/my-en/cloud/watson-natural-language-understanding.
[62] Y. Malheiros. *Access Senticnet Api Using Python*. Accessed: Sep. 13, 2019. [Online]. Available: https://pypi.org/project/senticnet/.
[63] GeoNames. *Geonames Web Services Documentation*. Accessed: Sep. 10, 2019. [Online]. Available: https://www.geonames.org/export/web-services.html
[64] F. Chollet. *Embedding Layer*. Accessed: Oct. 5, 2019. [Online]. Available: https://blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html
[65] F. Chollet. *Embedding layer*. Accessed: Oct. 5, 2019. [Online]. Available: https://keras.io/api/layers/core_layers/embedding/
[66] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," 2016, *arXiv:1606.01781*. [Online]. Available: http://arxiv.org/abs/1606.01781
[67] C. Olah. *Understanding LSTM Networks*. Accessed: Feb. 20, 2020. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
[68] W. L. Lim, C. C. Ho, and C.-Y. Ting, "Tweet sentiment analysis using deep learning with nearby locations as features," in *Computer Science Technology*. Singapore: Springer, 2020, pp. 291–299.
[69] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep convolution neural networks for Twitter sentiment analysis," *IEEE Access*, vol. 6, pp. 23253–23260, 2018.
[70] R. DeJordy and D. Halgin, *Introduction to Ego Network Analysis*. Boston MA, USA: Boston College Winston Center for Leadership & Ethics, 2008.
[71] K. Sugiyama, *Graph Drawing and Applications for Software and Knowledge Engineers*. Singapore: World Scientific, 2002.
[72] W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, Austin, TX, USA, 2010, pp. 51–56.
[73] T. E. Oliphant, *A Guide to NumPy*, vol. 1. New York, NY, USA: Trelgol, 2006.
[74] F. Chollet. (2015). *Keras*. [Online]. Available: https://github.com/fchollet/keras
[75] M. Abadi. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.Software available from tensorflow.org
[76] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[77] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances Neural Information Processing system*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, Eds. New York, NY, USA: Curran Associates, 2017, pp. 4765–4774.
[78] T. Hermosilla, J. Palomar-Vázquez, Á. Balaguer-Beser, J. Balsa-Barreiro, and L. A. Ruiz, "Using street based metrics to characterize urban typologies," *Comput., Environ. Urban Syst.*, vol. 44, pp. 68–79, Mar. 2014.
[79] Keras Team. *Merge Layers*. Accessed: Mar. 7, 2020. [Online]. Available: https://keras.io/layers/merge/

WEI LUN LIM is currently pursuing the M.Sc. degree in information technology by research with the Faculty of Computing and Informatics, Multimedia University. His research interests include machine learning and deep learning.

CHIUNG CHING HO (Senior Member, IEEE) is a Senior Lecturer with the Faculty of Computing and Informatics, Multimedia University. His research interests include location intelligence, biometrics, action recognition, and text mining.

CHOO-YEE TING is an Associate Professor with the Faculty of Computing and Informatics, Multimedia University. He is an active researcher in the areas of Bayesian networks, location analytics, and health insurance analytics.