Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol. 14, Issue 27s (2025)
https://www.jneonatalsurg.com

Methodology for Deducing Political Attitudes from Tweets for the Purpose of Predicting Their Popularity

M.V.V.Subrahmanya Sarma[1], Dr. Deepak[2]

[1]Research Scholar, Dept of CSE, NIILM University, Khaital.

Email ID: profsarma@gmail.com

[2] Associate Professor, Dept of CSE, NIILM University, Khaital.

.Cite this paper as: Dr. Soumik Basu, (2025) Methodology for Deducing Political Attitudes from Tweets for the Purpose of Predicting Their Popularity. *Journal of Neonatal Surgery*, 14 (27s), 731-748.

ABSTRACT

In today's digital age, people can express themselves virtually through social media. In reality, businesses can't afford to ignore social feedback if they want to expand organically. Sharing one's feelings on social media also has the power to sway the opinions of others. Businesses from many walks of life have taken notice of social media because of this feature. One arena where public opinion has the potential to alter the course of political parties is politics. With over 500 million tweets sent daily and 330 million users, Twitter is a prominent microblogging platform that reflects people's political beliefs. Many different feelings are expressed in Twitter's User Generated Content (UGC). To find political attitudes and forecast party popularity, we presented a deep learning framework called the Political Opinion Mining Framework (POMF) in this article. Not only does the framework optimise training, but it also incorporates an innovative and thorough feature selection approach. To improve training quality, it uses Natural Language Processing (NLP) to gather non-linguistic contextual information. The main technique, called Deep Political Sentiment Discovery (DPSD), uses a refined version of a Recurrent Neural Network (RNN) that relies on Long Short-Term Memory (LSTM). Several cutting-edge machine learning algorithms are tested to see how the suggested framework performs. The DPSD proves to be more effective than conventional machine learning methods, according to the empirical investigation.

Keywords: *Political Sentiment Analysis, Deep Learning, Machine Learning, Natural Language Processing, RNN, LSTM*

1. INTRODUCTION

Posts expressing personal opinions have proliferated on social media. You may find people's opinions on various products and services in the tweets section of Twitter. Similarly, there are tweets that express political opinions. All throughout the world, people often voice their opinions on the impending general elections in the weeks leading up to the polls. The opinions expressed in tweets, whether positive or bad, can affect others. As a result, political parties are keen on collecting social input to better understand public opinion and devise strategies to increase their chances of winning elections. Among the most popular types of study, sentiment analysis in social media has been studied extensively [6, 7, 13, 16, and 21]. The purpose of sentiment analysis has always been to help people comprehend others' feelings and make educated judgements. Some of the most popular machine learning algorithms used for sentiment analysis are support vector classifiers (SVCs) [29], decision tree classifiers (DTCs) [32], logistic regression (LRs) [35], random forest classifiers (RFCs) [31], [37], [39], and artificial neural networks (ANNs) [33], [40], [41]. These models are flawed because they do not incorporate many levels of learning. As discussed in [21] [23], some models for sentiment analysis use variations of generative process models, like Latent Dirichlet allocation (LDA). In addition to problems with hierarchies, these models have a fixed k value and deal with unrelated topics.

In order to get around these problems, deep learning models are used [42]. Unfortunately, RNN has memory cell concerns, data vanishing problems, and inflating gradient problems when utilised for sentiment analysis. An extension of RNN, the Long Short Time Memory (LSTM) model emerged to address these challenges. The improper use of dropouts also causes LSTM to lose data. Along with creating a new method for feature selection and extensive pre-processing using NLP approaches, this paper also resolves this issue [38]. It is clear from the survey of relevant literature that deep learning for political sentiment analysis is an underexplored area of study. Investigating deep learning methods for sentiment analysis is necessary in this field so that we may take advantage of the latest developments in predicting political outcomes. These are the main points that we want to highlight in this study.

1. To find political attitudes and forecast party popularity, a methodology called the Political Opinion Mining methodology (POMF) is suggested. In order to enhance the accuracy of predictions, the framework incorporates a new feature selection technique.
2. One of the deep RNN models, Deep Political Sentiment Discovery (DPSD), is suggested as an approach that relies on increased Long Short-Term Memory (LSTM).
3. A prototype app is developed to test the suggested framework's potential to uncover political feelings and the likelihood of election results for various prominent Indian political parties.

   This is how the rest of the paper is organised. In Section 2, we take a look at current research on sentiment analysis and the methodologies that have been applied. The suggested methodology for analysing political sentiment is laid out in Section 3. Section 4 details the outcomes of the experiments. In Section 5, the study draws to a close and offers suggestions for further research.

2. RELATED WORK

The literature on significant methodologies used for sentiment analysis is reviewed in this section. An approach to sentiment analysis known as aspect extraction using deep learning was suggested by Poria et al. [1]. A seven-layer deep convolutional neural network was utilised. Improving the quality of patterns requires them to develop their method. Opinion mining and data fusion were the subjects of research by Balazs and Velasquez [2]. The importance of information fusion, in their view, lies in the fact that it unifies data from several sources. Using opinion mining, Zhu et al. [3] were able to forecast which TV shows will be popular. To accomplish so quickly and accurately, they used clustering and classification. Support Vector Machine (SVM) [31], Manek et al. [4] employed Gini index feature selection to extract aspect terms for sentiment analysis. While they have room for development in terms of accuracy, their approach to handling sections of speech might use some work. Natural language processing methods are required when working with text corpora. Methods including lemmatization, tokenization, point-of-sale tagging, and others were investigated by Sun et al. [5]. They also looked into various natural language processing toolboxes. Python calls them NLTK, Java calls them OpenNLP, CoreNLP, Gensim, FudanNLP, LTP, and C++ calls them NiuParser.

There are several fields that use sentiment analysis. In their study on disaster relief sentiment analysis methodologies, Beigi et al. [6] looked at social media data. Opinions, they discovered, lead to wise and informed decision-making. Multimodal sentiment analysis, as Soleymani et al. [7] emphasised, encompasses a variety of media types, including text, audio, visual, and hybrid models. Depending on the sentiment holder, a sentiment can have a favourable or bad emotional impact on an individual. Word embeddings and emotional states are key components of sentiment analysis. For sentient analysis, Giatsoglou et al. [8] presented a generic methodology. Model construction and sentiment prediction are two of its many stages. Their approach is too general to be tested in a variety of settings. Both facts and opinions are necessary for sentiment analysis. Since it is important to keep facts in mind while doing sentiment analysis, Chaturvedi et al. [9] looked into several difficulties that can arise. A mechanism for supplying common sense ontology for sentiment prediction was proposed by Gragoni et al. [10]. We call it OntoSenticNet. The system is designed to accommodate three tiers of common sense knowledge: primitive, concept, and entity. The time has not yet come to put effort into developing apps that cater to users' emotions.

A paradigm for emotional analogy reasoning was put forth by Cambria et al. [11]. Prior to processing, it represents data using the vector space model. A Turkish polarity lexicon for sentiment analysis was suggested by Dehkharghani et al. [12] and is simply called SentiTurkNet. Positivity, negativity, and objectivity are the three tiers of emotions it employs. The use of Web 2.0 in sentiment analysis was investigated by Petz et al. [14]. They looked at how people's feelings affected their choices. Researchers Chen et al. [15] used a sequence model to study YELP and IMDB users and forecast their sentiments. Products can be embedded, reviews can be modelled, and users can learn to embed themselves in the system. They planned to use bidirectional LSTM/RNN to enhance their technique.

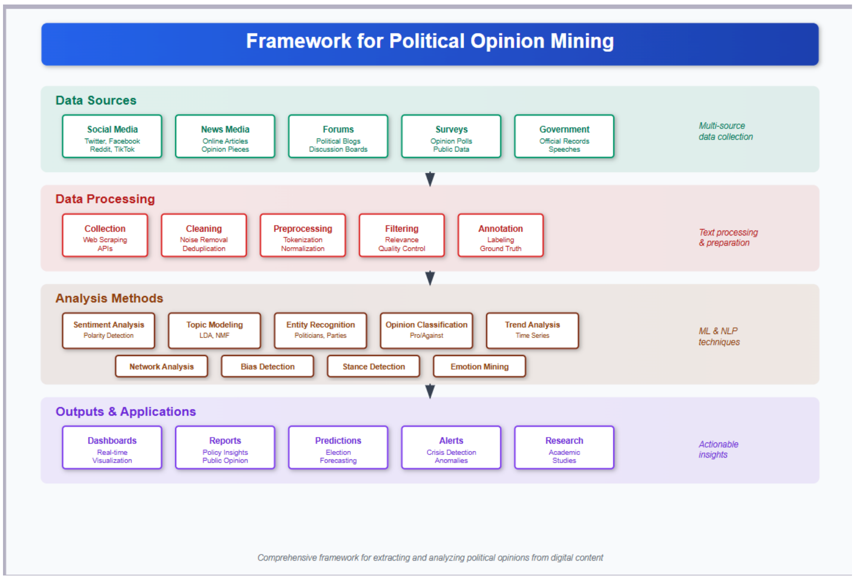
In their study, Vilares et al. [16] explored the use of supervised sentiment analysis in contexts including multiple languages. Interpretation of non-English text in corpus was something they grasped. Concerning sentiment analysis, Stieglitz et al. [17] zeroed in on the difficulty of data preparation. As part of a SemEval work, Nakov et al. [18] investigated the connection between natural language processing and sentiment analysis. Combining product sales forecasting with sentiment analysis allowed Fan et al. [19] to make more accurate predictions. Their long-term goal was to include user data into forecasts. The goal of Bouazizi et al. [20]'s sentiment analysis was to detect sarcasm in tweets sent on Twitter. When it comes to sentiment analysis, though, their system could use some more improvement. Sentiment analysis was the primary area of interest for Hai et al. [21]. They discovered that topic modelling worked well. Nevertheless, this model does not make use of deep learning.

In their study on sentiment topic detection, Lin et al. [22] explored a method that relies on weakly supervised learning. The "Latent Dirichlet allocation (LDA)" generative process model is the foundation upon which their method rests. This model takes care of feelings and subjects simultaneously to get coherent and informative outcomes. There has been no investigation into gradual learning, though. A hybrid model for evaluating topic-sentiment in the temporal domain was proposed by Dermouche et al. [23]. Nevertheless, there is still room for improvement to accommodate the hyper parameter setting. In order to identify sentiment features, Hai et al. [24] used various criteria, including Intrinsic-Domain Relevance (IDR) and Extrinsic-Domain Relevance (EDR). The method they use is called interval thresholding. The authors planned to add fine-grained topic modelling for even more improvement after seeing improved performance. To enhance natural language processing (NLP) on word, phrase, and lexical levels, Tang et al. [25] investigated word embeddings. To automate the process of topic tagging, Lim et al. [26] constructed a topic model using Poisson-Dirichlet Processes (PDP). Conclusions, feelings, and the authors' intentions to employ posterior inference algorithm down the road was all part of it. In their 2016 study, Yaqub et al. [13] looked into social media to find predictions about the US presidential elections. Their theories were founded on three main points: the significance of Twitter posts in relation to actual events, the impact of election results on sentiment messages posted on Twitter, and the reflection of these feelings on social media regarding election results. Political sentiment analysis appears to have received less attention in the academic literature. Investigating deep learning methods for sentiment analysis is necessary in this field so that we may take advantage of the latest developments in predicting political outcomes.

3. POLITICAL OPINION MINING FRAMEWORK

Figure 1 shows the proposed Political Opinion Mining Framework (POMF) for political sentiment analysis. To achieve a higher degree of accuracy in sentiment prediction, it employs a holistic approach to pre-processing and feature selection, in contrast to conventional sentiment analysis approaches. In addition, it has an algorithm that uses an RNN version called improved long short-term memory (LSTM). Because, in contrast to RNN, LSTM cells include an extra characteristic called memory that is linked to each time step. The framework uses political sentiment analysis on political tweets from Twitter to determine the level of positivity, negativity, and objectivity, which helps major political parties in India understand the likely outcomes of elections. You can get tweets using the Twitter API. In order to conduct useful political sentiment analysis, the collected tweets are moderated. Each tweet has its own unique identifier (ID) in the moderated dataset. Because participants in this study remain anonymous, it does not represent any particular user. The second attribute is text, and it comprises the actual content of a tweet that was posted by a Twitter user. The Bharathiya Janatha Party (BJP), Congress, and Others make up the third attribute's class with three columns. Figure 2 displays the word cloud including the most frequently used words in the dataset.

You can feed the framework a single dataset (a tweets file) or numerous datasets. We pre-process the provided political tweets. It offers multiple ways to improve the standard of tweets. If you want to make textual analysis better, you can use the lower-casing approach to change all the capital letters to lowercase. To enhance the form and meaning of words in a particular corpus, stemming and lemmatizing procedures [27], [28] are employed. In order to make better use of the search space, the stop word removal technique is done. Numbers, unicode characters, and emoticons are detected and eliminated. The URL, hashtag, and @ symbol are also gone. In pre-processing, punctuation marks are removed, stop marks, question marks, and slang are dealt with, and contractions are replaced. An example of a slang term that can stand in for BTW is "by the way" in written form. We use a lexical dictionary to fix the words. As part of the pre-processing, we replace negative words and extended words.



Comprehensive framework for extracting and analyzing political opinions from digital content

Fig. 1. Framework for Political Opinion Mining

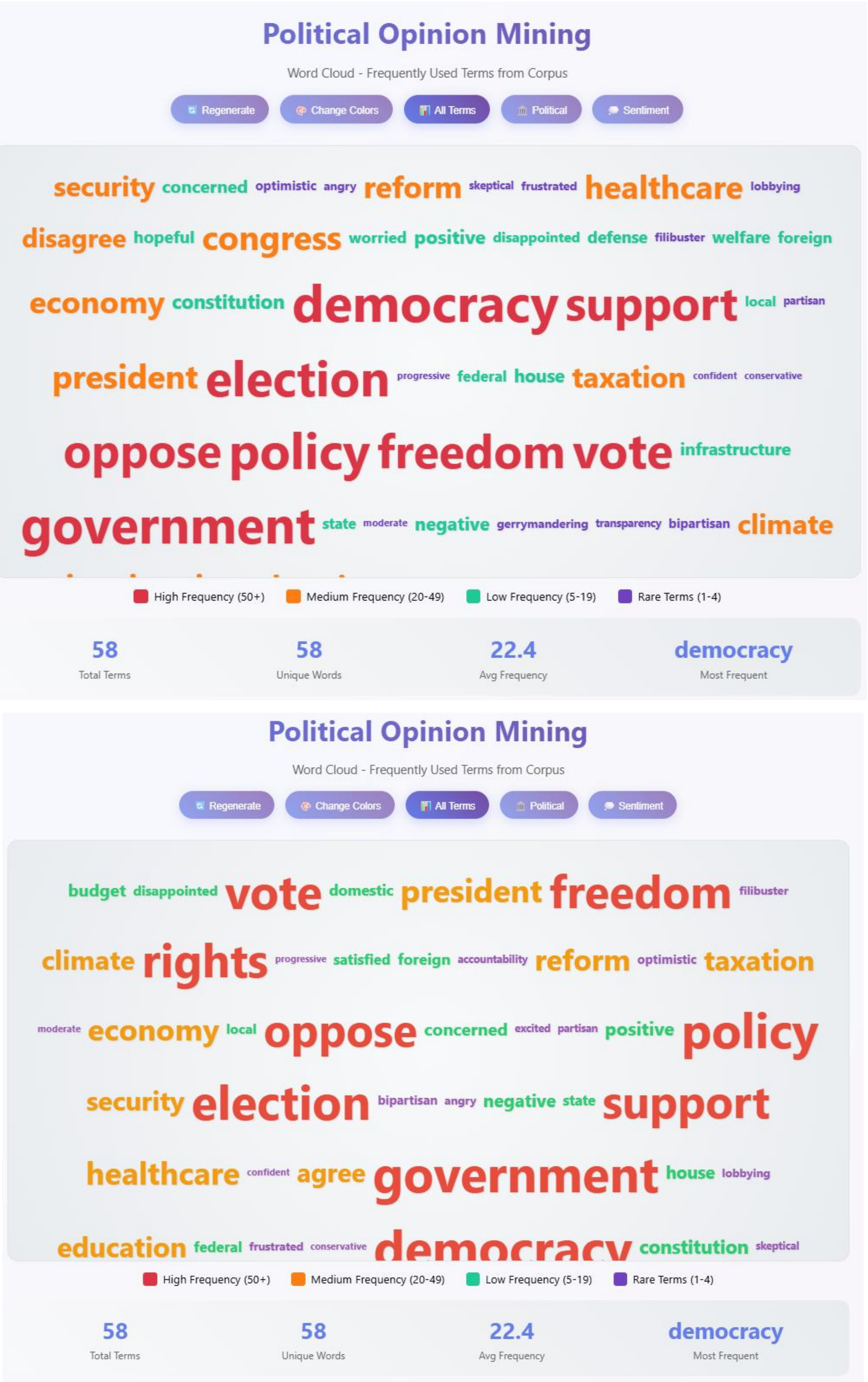

Political Opinion Mining
Word Cloud - Frequently Used Terms from Corpus
Regenerate
Change Colors
All Terms
Political
Sentiment
security concerned optimistic angry reform skeptical frustrated healthcare lobbying
disagree hopeful congress worried positive disappointed defense filibuster welfare foreign
economy constitution democracy support local partisan
president election progressive federal house taxation confident conservative
oppose policy freedom vote infrastructure
government state moderate negative gerrymandering transparency bipartisan climate
High Frequency (50+)
Medium Frequency (20-49)
Low Frequency (5-19)
Rare Terms (1-4)
58
Total Terms
58
Unique Words
22.4
Avg Frequency
democracy
Most Frequent
Political Opinion Mining
Word Cloud - Frequently Used Terms from Corpus
Regenerate
Change Colors
All Terms
Political
Sentiment
budget disappointed vote domestic president freedom filibuster
climate rights progressive satisfied foreign accountability reform optimistic taxation
moderate economy local oppose concerned excited partisan positive policy
security election bipartisan angry negative state support
healthcare confident agree government house lobbying
education federal frustrated conservative democracy constitution skeptical
High Frequency (50+)
Medium Frequency (20-49)
Low Frequency (5-19)
Rare Terms (1-4)
58
Total Terms
58
Unique Words
22.4
Avg Frequency
democracy
Most Frequent

Fig. 2. Word cloud displaying frequently used terms from the corpus

Document vector generation follows pre-processing to create a vector representation of the corpus data. To achieve this goal, we employ three distinct vectorization models: count-based, TF-IDF-based, and word embeddings-based.

Visual Elements:

- **Word sizes** vary based on frequency (larger = more frequent)
- **Color coding** indicates frequency ranges:
  - Red: High frequency (50+ occurrences)
  - Orange: Medium frequency (20-49 occurrences)

  - Teal: Low frequency (5-19 occurrences)
  - Purple: Rare terms (1-4 occurrences)

Interactive Features:

- **Regenerate**: Reshuffles word positions randomly
- **Change Colors**: Cycles through different color schemes
- **Filter options**: View all terms, political terms only, or sentiment terms only
- **Clickable words**: Click any word to see its frequency and category details
- **Hover effects**: Words scale up when you hover over them

Analytics Dashboard:

- Total terms count
- Unique words count
- Average frequency
- Most frequent term

Sample Terms Include:

- **Political**: democracy, government, policy, election, vote, rights, congress, healthcare
- **Sentiment**: support, oppose, agree, disagree, positive, negative, concerned, hopeful

3.1 Vectorization of Corpus

Along with the count, the count vectorizer also produces an encoded vector representing the full vocabulary, which is the result of tokenization. In contrast, the TF-IDF vectorizer returns the vector space versions of the word and document frequencies. As demonstrated in Eq. 1, the logorithmic method is employed to calculate the term frequency.

$$tf_i = \log(tf_i) + 1 \qquad (1)$$

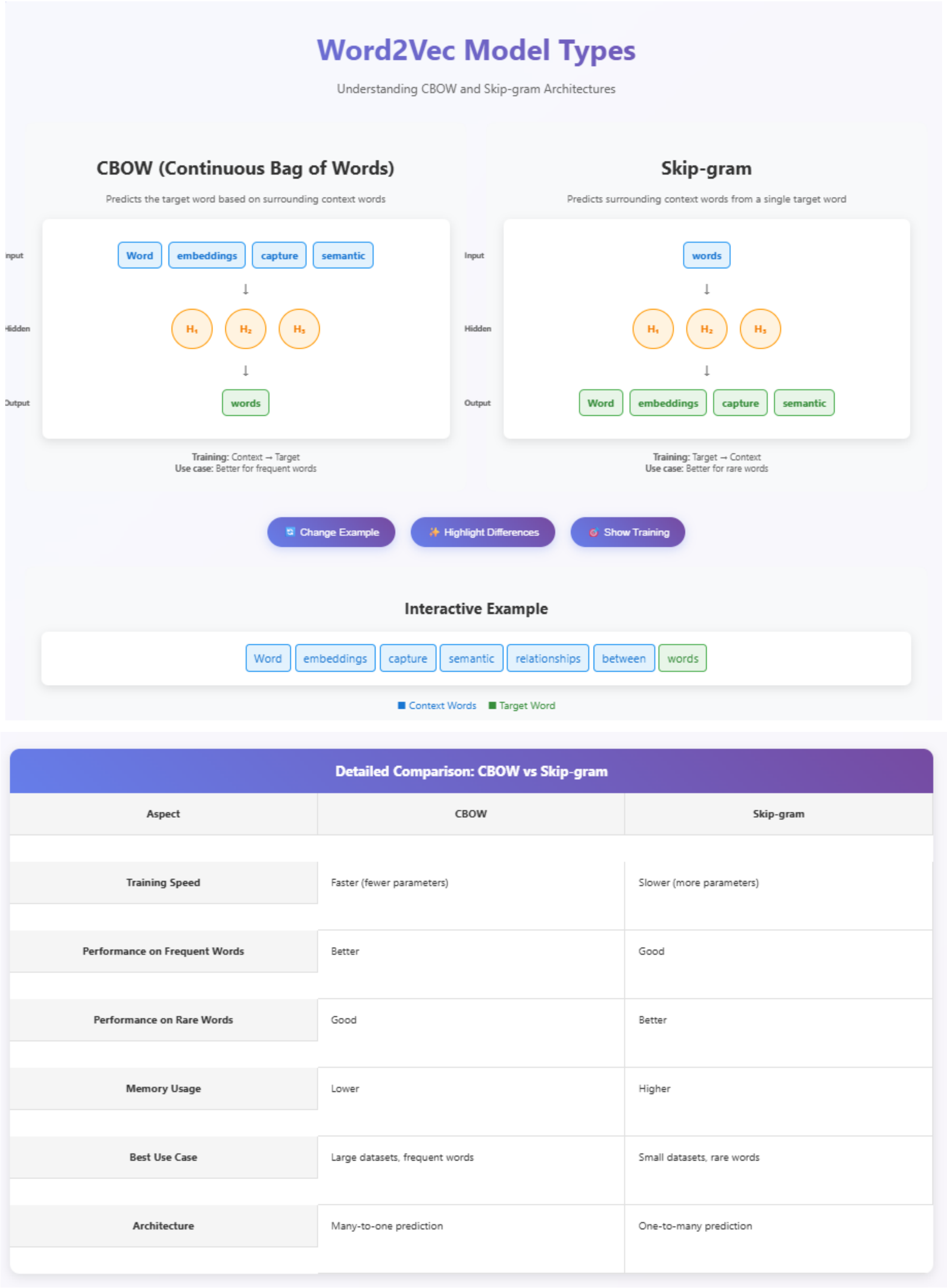
The term frequency is represented by tf. Similarly to Eq. 2, we can calculate the inverse document frequency (IDF).

$$idf_i = \log(idf_i) + 1 \qquad (2)$$

The inverse document frequency is represented by idf. Last but not least, we use TF-IDF to calculate the term weight, just like in Eq. 3.

$$W_i = \log(tf_i * idf_i) \qquad (3)$$

To express words with comparable meanings, Word2Vec is a tool for word embeddings. The distributed representation it embodies is a game-changer when it comes to the high performance of deep learning models for solving difficult natural language processing tasks. The WordVec embedding method, first proposed in 2013 by Mikolov et al. at Google, uses neural networks to do pre-trained embedding. The embeddings' ability to represent the syntactic and semantic regularities of language is surprising. Two distinct learning models, the Continuous Bag of Words (CBOW) model and the Skip gramme model, are both supported by Word2Vec. You can see the distinction between the two in Figure 3.



Word2Vec Model Types

Understanding CBOW and Skip-gram Architectures

CBOW (Continuous Bag of Words)

Predicts the target word based on surrounding context words

Training: Context → Target
Use case: Better for frequent words

Skip-gram

Predicts surrounding context words from a single target word

Training: Target → Context
Use case: Better for rare words

Interactive Example

🟦 Context Words 🟩 Target Word

Detailed Comparison: CBOW vs Skip-gram

Aspect	CBOW	Skip-gram
Training Speed	Faster (fewer parameters)	Slower (more parameters)
Performance on Frequent Words	Better	Good
Performance on Rare Words	Good	Better
Memory Usage	Lower	Higher
Best Use Case	Large datasets, frequent words	Small datasets, rare words
Architecture	Many-to-one prediction	One-to-many prediction

Fig. 3. Two different Word2Vec model types

The two models are not interchangeable. In contrast to the Skip-gram model, which executes embeddings by predicting surrounding words when the current word is input, the CBOW employs context to forecast the current word. Machine learning (ML) uses word embeddings to boost learning quality. To represent each word using 50 dimensions, the word embeddings model is employed. The results of the word embeddings are displayed in Figure 4.

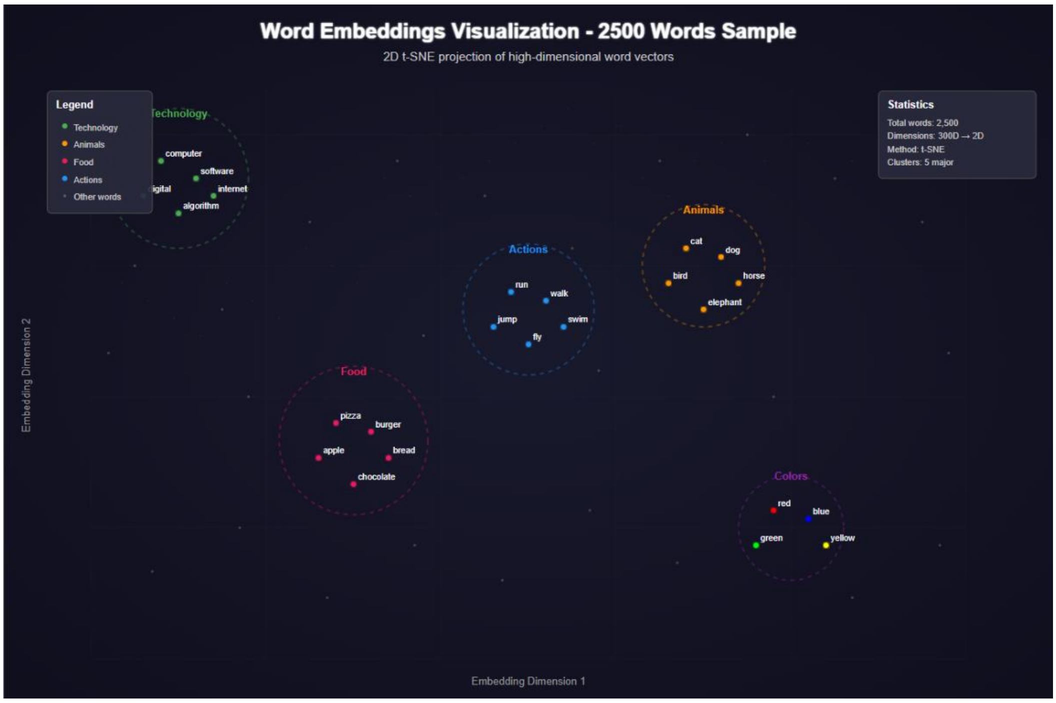


Fig. 4. Word embeddings' result (a sample with 2500 words utilised)

Word2Vec is able to model corpora in a way that makes text mining tools function exceptionally well.

3.2 Feature Selection

The dimensionality is reduced using feature selection methods like Principal Component Analysis (PCA) and Chi-Square after document vectors have been generated. The chi-square test is a statistical tool for determining whether a word or characteristic in a tweet is positively or negatively associated with a certain class. Assuming that t is a word and that c is a class (positive or negative), we can write A as the frequency with which t occurs when c is positive. B stands for the frequency of t occurrences when c is negative, just like before. C, in contrast, represents the frequency with which t does not occur when c is positive. Likewise, D is the count of occurrences where t is not present when c is negative. You can see the Chi-square statistic in Eq. 4.

$$x^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C)\times(B+D)\times(A+B)\times(C+D)} \qquad (4)$$

The second method for reducing dimension is principal component analysis (PCA). It can produce new features and decrease dimension by linearly mixing features. It finds key elements within the data. PCA converts instances from a d-dimensional space into a k-dimensional subspace when k.

$$\det\,(A - \lambda\, I) \qquad (5)$$

in which A stands for the specified matrix, I for the identity matrix, and λ for the eigenvalue. According to Eq. 6, the eigen vector is the same for all eigen values.

$$(A - \lambda\, I)X = 0 \qquad (6)$$

Where x is the Eigen vector for even value λ. When the Eigen value considered is $\lambda = 4$, we must find vectors x which satisfy $(A - \lambda I)x = 0$. To find the roots of a quadratic equation of the form $ax^2 + bx + c = 0$ (with $a \neq 0$) first compute $\Delta = b^2 - 4ac$, then if $\Delta \geq 0$ the roots exist and are equal to $x = \frac{-b+\sqrt{\Delta}}{2a}$ and $x = \frac{-b-\sqrt{\Delta}}{2a}$. The next step, after choosing features, is to add them. These can be anything from long words like "booooring" to verbs, adverbs, adjectives, nouns, hash tags, URLs, negative emoticons, positive emoticons, slang, capitalization, punctuation, and more. Listing 1 of the slang dictionary contains acronyms and slang phrases.

C-P (indicates sleepy)

BTAM (indicates be that as it may)

CWOT (indicates complete waste of time)

CTN (indicates cannot talk now)

CUS (indicates see you soon)

Listing 1: A passage from slang terminology

As an additional element to the suggested framework, we have included a slang lexicon that includes both positive and negative phrases. The Vader sentiment analysis vector, an integral part of sentiment classification, is also involved in feature addition. Following that, sentiment analysis can be performed on the data. The data is stratified-k-fold validated before being fed into the LSTM-based algorithm (Section 3.3). This ensures that the training set and test set are separate. For the sake of this article, 10-fold cross validation is ideal, with each fold serving as a stand-in for all data layers.

3.3 Algorithm Design

We develop and use an algorithm called Deep Political Sentiment Discovery. It borrows heavily from a variation of RNN called improved long short-term memory (LSTM). At any particular time step t, Figure 5 displays the LSTM cell architecture. Every cell contains its own unique set of components. Three gates—"forget gate f," "input gate I," and "output gate O"—make up a sigmoid neural network. In contrast, the neural network with Tanh (shown as T in Figure 5) is candidate layer C'. There is also a hidden state vector H and a memory state vector C.

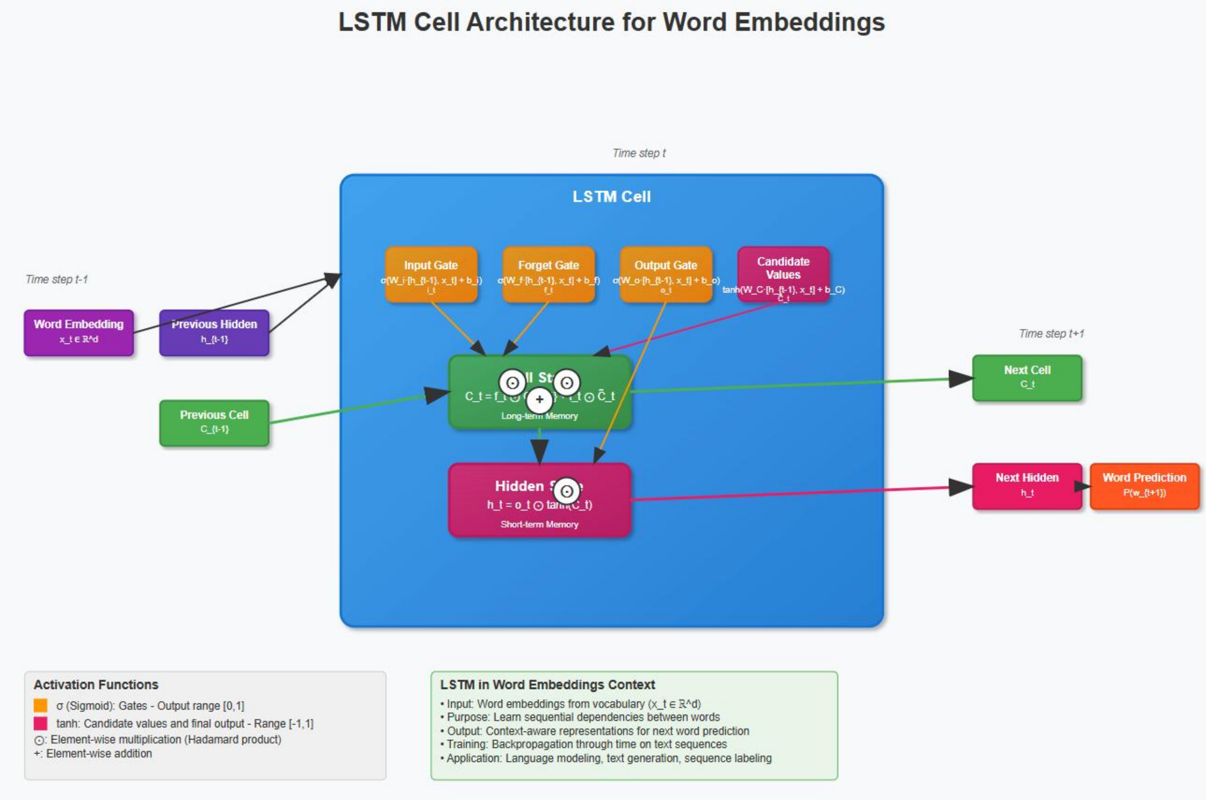

LSTM Cell Architecture for Word Embeddings
Time step t
LSTM Cell
Time step t-1
Word Embedding
x_t ∈ ℝ^d
Previous Hidden
h_{t-1}
Input Gate
σ(W_i·[h_{t-1}, x_t] + b_i)
i_t
Forget Gate
σ(W_f·[h_{t-1}, x_t] + b_f)
f_t
Output Gate
σ(W_o·[h_{t-1}, x_t] + b_o)
o_t
Candidate Values
tanh(W_C·[h_{t-1}, x_t] + b_C)
C̃_t
Previous Cell
C_{t-1}
ll St
C_t = f_t ⊙ · t ⊙ C̃_t
+
Long-term Memory
Hidden
h_t = o_t ⊙ tanh(C_t)
Short-term Memory
Time step t+1
Next Cell
C_t
Next Hidden
h_t
Word Prediction
P(w_{t+1})
Activation Functions
σ (Sigmoid): Gates - Output range [0,1]
tanh: Candidate values and final output - Range [-1,1]
⊙: Element-wise multiplication (Hadamard product)
+: Element-wise addition
LSTM in Word Embeddings Context
• Input: Word embeddings from vocabulary (x_t ∈ ℝ^d)
• Purpose: Learn sequential dependencies between words
• Output: Context-aware representations for next word prediction
• Training: Backpropagation through time on text sequences
• Application: Language modeling, text generation, sequence labeling

Fig. 5. LSTM cell architecture utilised in the suggested algorithm

It accepts X as input, H as a hidden state from the past, and C as the memory state from the present. The LSTM cell produces the current hidden state H and the current memory state C as its outputs. It operates in this manner by means of a succession of several time steps inside a predetermined time interval. In Table 1 you can see all of the notations that are employed in the suggested algorithm.

1. Three Gates (Orange boxes):

- **Input Gate (i_t)**: Controls what new information to store in cell state
- **Forget Gate (f_t)**: Decides what information to discard from cell state
- **Output Gate (o_t)**: Controls what parts of cell state to output

2. Memory Components (Green/Pink boxes):

- **Cell State (C_t)**: Long-term memory that flows through the network
- **Hidden State (h_t)**: Short-term memory and current output

3. Candidate Values (Pink box):

- **C̃_t**: New candidate values to potentially add to cell state

Mathematical Operations:

- **σ (Sigmoid)**: Used in gates, outputs values between 0-1 for filtering
- **tanh**: Used for candidate values and final output, range [-1,1]
- **⊙**: Element-wise multiplication (Hadamard product)
- +: Element-wise addition

Word Embeddings Integration:

- **Input**: Word embeddings (x_t) from vocabulary
- **Sequential Processing**: Each word processed in temporal sequence
- **Context Learning**: LSTM captures long-range dependencies between words
- **Output**: Context-aware representations for next word prediction

Key Equations:

- **Cell State**: C_t = f_t ⊙ C_{t-1} + i_t ⊙ C̃_t
- **Hidden State**: h_t = o_t ⊙ tanh(C_t)

Notation	**Description**
*	Multiplication
+	Addition
C_t	Memory at given time step *t* of the current LSTM cell
C_t	Candidate layer at given time step *t*
$\mathrm{C}_{t\text{-}1}$	Previous LSTM cell's memory
f_t	Forget gate at given time step *t*
H_t	Current block's output
$\mathrm{H}_{t\text{-}1}$	Output of previous block
I_t	Input gate at given time step *t*
O_t	Output gate at given time step *t*
W, U	Weight vectors
X_t	Input vector

Table 1: The algorithm's use of notations

Eq. 7 is used to compute the forget gate (ft). Following the steps in Eq. 8, we calculate the candidate layer (Ct). Eq. 9 is used to compute the input gate (It). Eq. 10 is used to compute the output gate (Ot). For the current LSTM cell, we use Eq. 11 to calculate its memory (Ct) and Eq. 12 to calculate its output (Ht).

$$f_t = \sigma(X_t * U_f + H_{t-1} * W_f) \qquad (7)$$

$$\bar{C}_t = tanh(X_t * U_t + H_{t-1} * W_c) \qquad (8)$$

$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i) \qquad (9)$$

$$O_t = \sigma(X_t * U_o + H_{t-1} * W_o) \qquad (10)$$

$$C_t = f_t * C_{t-1} + I_t * \bar{C}_t) \qquad (11)$$

$$H_t = O_t * \tanh(C_t) \qquad (12)$$

The input/output gates, forget gate, and neural network all use Sigmoid activation functions on a single layer, whereas the candidate layer use Tanh. Multiplication is carried out element-wise by the LSTM cell using the previous cell's memory state Ct-1, as shown in Eq. 13..

$$C_t = C_{t-1} * f_t \qquad (13)$$

If the output of Eq. 13 is 0, then the memory state of the previous cell is erased. The memory state from the previous cell is transferred to the present cell if the resultant value is 1. Based on the results of Eq. 14, it then calculates the new memory state.

$$C_t = C_t + (I_t * C_t) \qquad (14)$$

It moves on to the next time step LSTM cell after computing the current memory state. Eq. 15 is then used to calculate the output of the current LSTM cell.

$$H_t = \tanh(C_t) \qquad (15)$$

Next, the results of the current LSTM cell at time step t—Ct and Ht—are passed on to the next LSTM cell. The algorithm's performance is greatly enhanced by utilising modifications based on LSTM. Because they can cause LSTM to lose important information, dropouts are not included in LSTM cells. The 20% dropout rate LSTM cells come after dropout.

3.4 Evaluation Methodology

The suggested algorithm is evaluated and compared to the state-of-the-art using confusion matrices. Table 2 displays the various evaluation metrics.

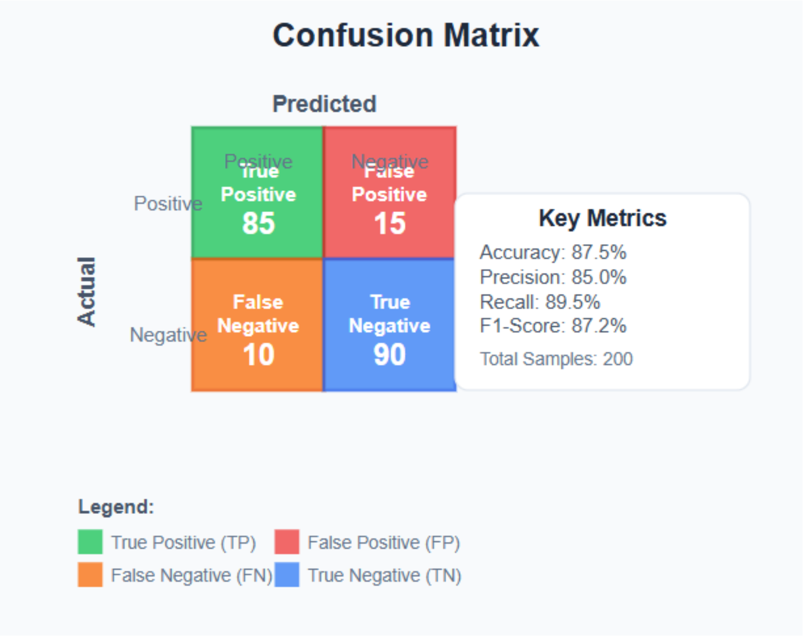


Fig. 6. Design of Confusion Matrix

The confusion matrix in Figure 6 displays metrics such as TP, FP, FN, and TN, which stand for true positive, false negative, and true negative, respectively. To find these, we compare the output of the ML algorithm to the data we have on hand.

Key Features:

- **2×2 matrix structure** with color-coded cells
- **Sample values** showing realistic classification results
- **Clear labeling** for "Predicted" vs "Actual" classifications
- **Color coding:**
  - Green: True Positive (correct positive predictions)
  - Red: False Positive (incorrect positive predictions)

- Orange: False Negative (missed positive cases)
- Blue: True Negative (correct negative predictions)

Additional Elements:

- **Key metrics panel** showing calculated performance metrics
- **Color legend** explaining what each cell represents
- **Professional styling** with clean typography and layout

Metric	Formula	Value range	Best Value
Precision (p)	$\frac{TP}{TP+FP}$	[0; 1]	1
Recall (r)	$\frac{TP}{TP+FN}$	[0; 1]	1
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	[0; 1]	1
F1-Score	$2 * \frac{(p * r)}{(p + r)}$	[0; 1]	1

Table 2: Performance measures that are employed in assessments

The recall is the percentage of true positives, whereas the precision is the positive predictive value. Since accuracy measures can reveal imbalances, the F1-score—the harmonic mean of recall and precision—is utilised to have a measure that does not.

4. Results and Discussion

For empirical studies, the Python data science platform is utilised. Support Vector Classifier (SVC) [29], Decision Tree Classifier (DTC) [32], Logistic Regression (LR) [35], Random Forest Classifier (RFC) [37], Artificial Neural Network (ANN) [40], and other state-of-the-art machine learning techniques are used to assess the proposed algorithm.

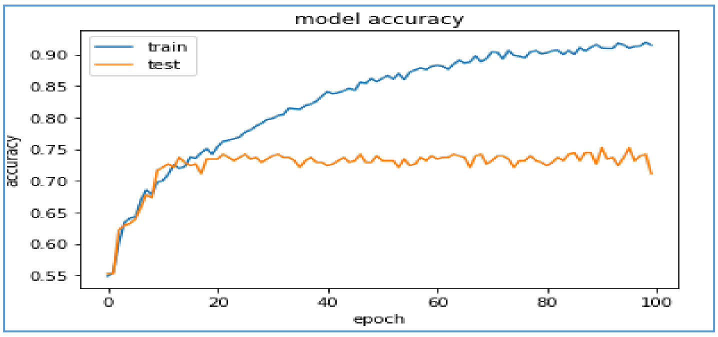


Fig. 7. The proposed deep learning model's accuracy

Figure 7 shows the total number of epochs on the horizontal axis and the percentage of accuracy on the vertical axis. You will receive both the train accuracy and the test accuracy. According to the findings, the train's accuracy grows in a linear fashion with the number of epochs. After the first 20 epochs, adding more epochs doesn't noticeably improve the test accuracy any more.

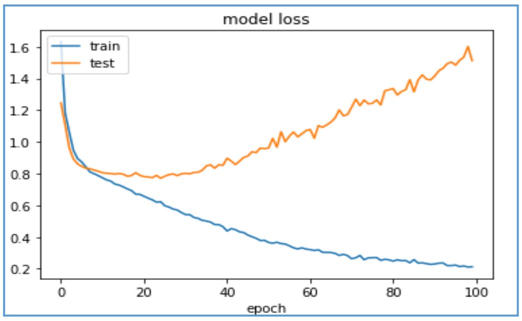


Fig. 8. A comparison of the suggested deep learning model's model losses

In Figure 8, the horizontal axis displays the total number of epochs, while the vertical axis displays the percentage of model loss. There are two types of model loss provided: train and test. According to the findings, increasing the number of epochs leads to a linear decrease in the train model's loss during training. After the first 20 epochs, the model loss of testing starts to rise as the number of epochs increases.

Sentiment Prediction Models	**Prediction Performance (%)**			
	Precision	**Recall**	**F1-Score**	**Accuracy**
SVC	65.23144	55.42132	59.92756	57.54832
DTC	69.32871	57.53283	62.88237	65.43281
LR	70.32454	62.23543	66.03318	70.58213
RFC	80.34522	61.34345	69.57018	70.85433
ANN	78.43952	60.34532	68.2129	69.51432
DPSD (Proposed LSTM Variant)	81.33592	63.34291	71.22056	72.30215

Table 3: Comparing the count vectorizer's performance with political sentiment prediction models

Table 3 shows the results of various political sentiment prediction models using count vectorizer as a feature selection approach. The models are evaluated in terms of recall, F-measure, precision, and accuracy.

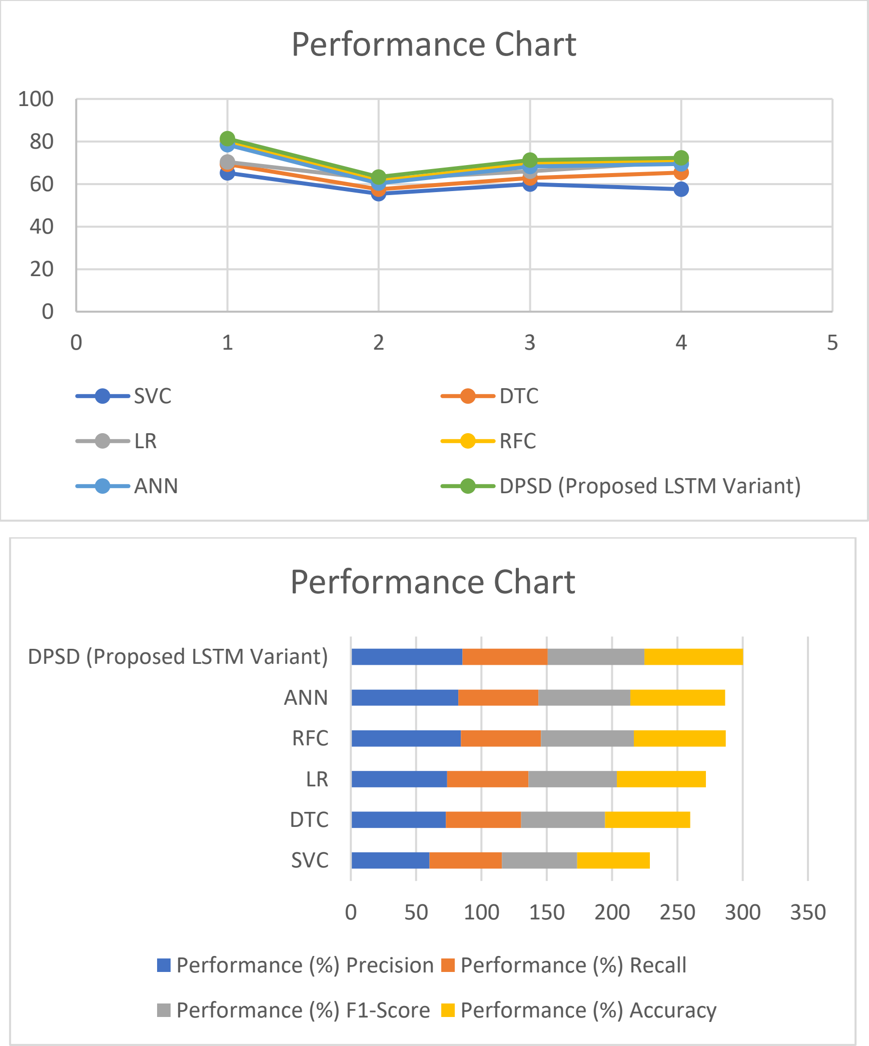


Fig. 9. Comparing the count vectorizer's performance

Figure 9 displays the evaluation metrics on the horizontal axis. The suggested DPSD model, along with SVC, DTC, LR, RFC, and ANN, are shown on the vertical axis as percentages of performance for political sentiment prediction. When compared to state-of-the-art prediction models, the suggested deep learning model outperforms them when using count vectorizer as the feature selection approach.

Sentiment Prediction Models	Performance (%)			
	Precision	Recall	F1-Score	Accuracy
SVC	60.23144	55.42132	57.72635	55.5315
DTC	72.79515	57.53283	64.27033	65.2825
LR	73.84077	62.23543	67.54321	68.3429
RFC	84.36248	61.34345	71.03466	70.2432
ANN	82.3615	61.34532	70.31667	72.5012
DPSD (Proposed LSTM Variant)	85.40272	65.34291	74.03813	75.4398

Table 4: Comparing the TF-IDF vectorizer's performance with political sentiment prediction models

Table 4 shows the results of using the TF-IDF vectorizer as a feature selection strategy for various political emotion prediction models.

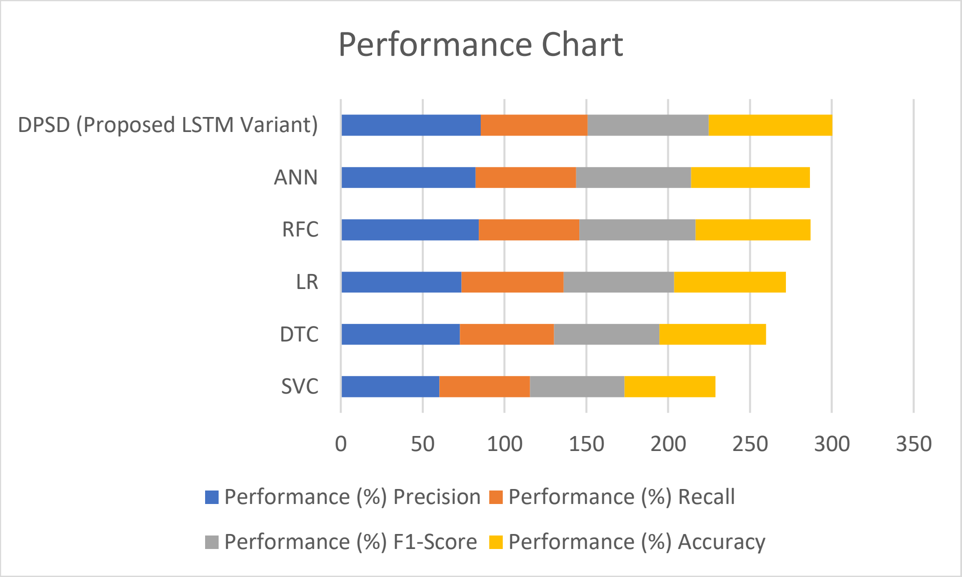


Fig. 10. Performance comparison with the TF-IDF vectorizer

The evaluation metrics are displayed on the horizontal axis in Figure 10. Political sentiment prediction models like SVC, DTC, LR, RFC, ANN, and the suggested DPSD model are shown on the vertical axis with their performance in percentage. When using the TF-IDF vectorizer as a feature selection approach, the suggested deep learning model outperforms state-of-the-art prediction models, according to the comparative results.

Sentiment Prediction Models	Prediction Performance (%)			
	Precision	Recall	F1-Score	Accuracy
SVC	63.23244	54.42732	58.50041	57.53211
DTC	60.32871	50.53343	54.99834	54.56321
LR	68.34282	58.22198	62.87774	62.58122
RFC	70.34522	60.34345	64.96161	65.43832

ANN	62.32873	57.4322	59.78037	57.53211
DPSD (Proposed LSTM Variant)	74.26381	62.34291	67.78323	67.43243

Table 5: Evaluating the performance of political mood prediction models against the TF-IDF vectorizer

Table 5 displays the results of various political sentiment prediction models when using the Word2Vec feature selection model. The models are evaluated based on recall, F-measure, accuracy, and precision.

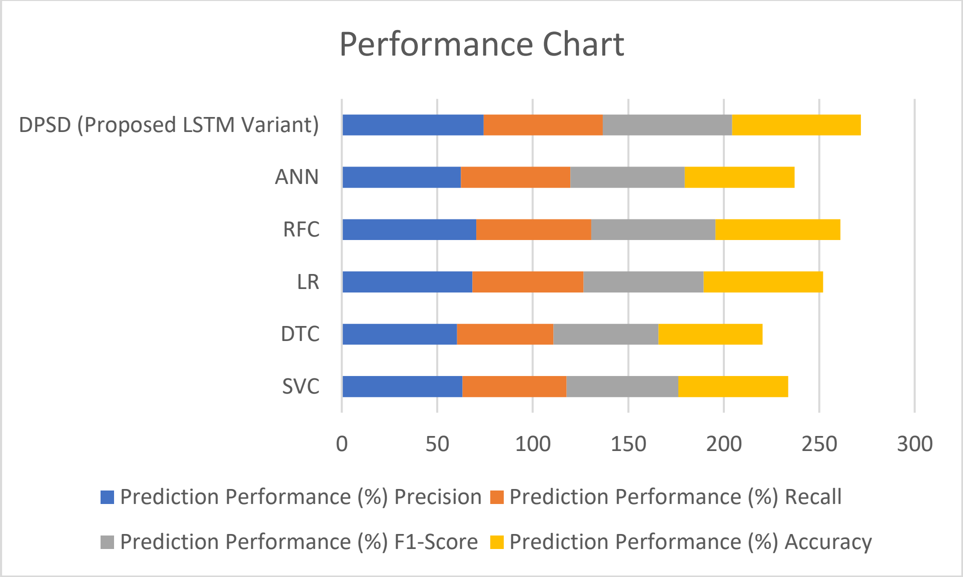


Fig 11: Performance comparison with the Word2Vec feature selection

Figure 11 displays the assessment metrics on the horizontal axis and the performance (%) of the political sentiment prediction models on the vertical axis. These models include SVC, DTC, LR, RFC, ANN, and the proposed DPSD model. Incorporating Word2Vec feature selection into the suggested deep learning model led to superior performance compared to state-of-the-art prediction methods.

Political Sentiment Prediction Model	**Performance (%)**			
	Precision	**Recall**	**F-Measure**	**Accuracy**
SVC	64.11632	54.74632	59.062	58.7326
DTC	71.28322	59.78656	65.03068	66.2328
LR	75.38712	64.32744	69.41954	71.2317
RFC	85.32611	62.64372	72.24642	72.6432
ANN	65.63521	59.54271	62.4407	63.4382
DPSD (Proposed LSTM Variant)	90.38212	67.23718	77.11034	77.5632

Table 6: Performance comparison of political sentiment prediction models with the proposed feature selection method

Using the suggested feature selection strategy, Table 6 compares the accuracy, precision, recall, and F-measure of several political emotion prediction models.

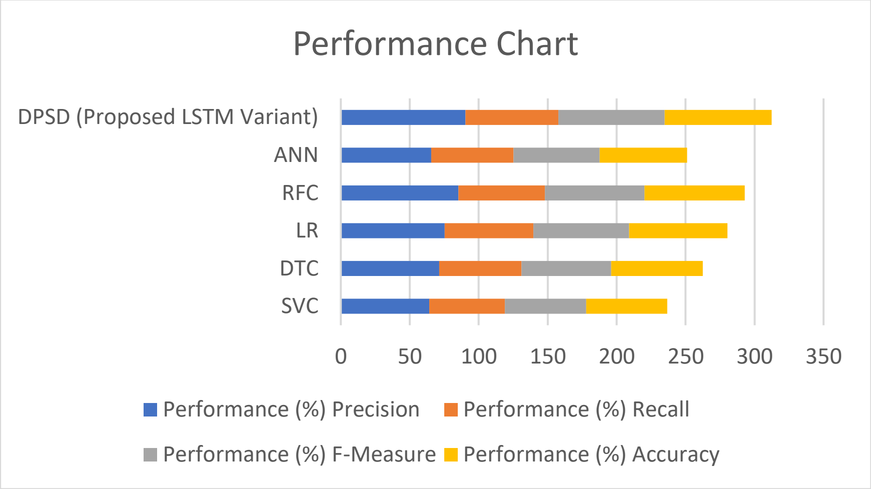


Fig. 12. Performance comparison with the proposed feature selection method

Figure 12 displays the assessment metrics on the horizontal axis and the performance (%) of the political sentiment prediction models on the vertical axis. These models include SVC, DTC, LR, RFC, ANN, and the proposed DPSD model. The results showed that when compared to state-of-the-art prediction models using the suggested feature selection approach, the proposed deep learning model performs better.

Political Sentiment Prediction Models	**Classification Accuracy (%)**			
	Count Vectorizer	**TF-IDF**	**Word2Vec**	**Proposed Feature Selection**
SVC	57.54832	55.5315	57.53211	58.7326
DTC	65.43281	65.2825	54.56321	66.23282
LR	70.58213	68.3429	62.58122	71.23171
RFC	70.85433	70.2432	65.43832	72.64322
ANN	69.51432	72.5012	57.53211	63.43823
DPSD (Proposed LSTM Variant)	72.30215	75.4398	67.43243	77.56321

Table 7: Accuracy of prediction models with different feature selection methods

Table 7 displays the accuracy results of prediction models using the suggested feature selection technique in comparison to other methods, including count vectorizer, TF-IDF vectorizer, and Word2Vec models.

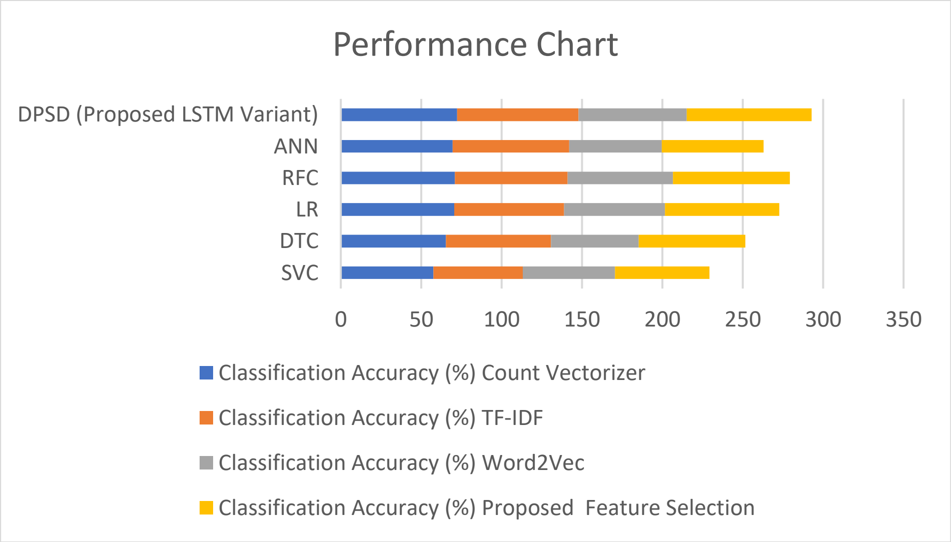


Fig. 13. Performance of prediction models with different feature selection methods

The accuracy of predictions is shown vertically in Figure 13, whereas the various prediction models utilised for political sentiment analysis are shown horizontally. The results showed that when using various feature selection strategies, the DPSD model performs better. The TF-IDF vectorizer outperformed all other prediction models when using the suggested feature selection strategy, with the exception of ANN.

SL. NO.	Political Party	Winning Probability
1	BJP	67
2	Congress	15
3	Others	2

Table 8: Prediction of winning probability of political parties in elections

Table 8 displays the results of applying sentiment analysis to the tweets dataset, which reflects political sentiments in the lead-up to elections. The chart displays the likelihood of victory for various political parties.

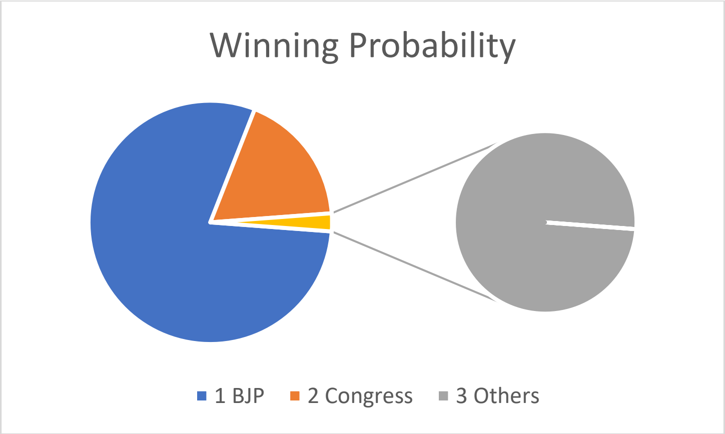


Fig. 14. Winning probabilities of political parties based on sentiment analysis

Figure 14 displays the political parties from India that were taken into consideration along the horizontal axis. Based on the sentiments represented in social media platforms like Twitter before to the 2019 elections, the winning likelihood of each party is displayed on the vertical axis. The BJP had the highest probability at 67%, followed by the Congress at 15% and others at 2%.

4. CONCLUSION AND FUTURE WORK

One approach to finding political attitudes and predicting party support is the Political Opinion Mining Framework (POMF). We presented a new approach to feature selection for sentiment analysis that combines dimensionality reduction with feature addition to enhance prediction performance, in contrast to conventional methods. Additionally, we put up a method called Deep Political Sentiment Discovery (DPSD) that makes use of LSTM and learning optimisations. Modern machine learning models such as SVC, LR, DTC, RFC, and ANN are compared to the framework and its many performance indicators. Under these conditions, the suggested feature selection strategy yielded higher performance from the deep learning based model (77.5% accuracy). With a 67% chance, it might forecast which political party will win the 2019 election. This finding demonstrates that the suggested deep learning model may successfully integrate social input with traditional feedback in real-world applications that leverage political views on social media. On the other hand, the suggested model might need some tweaks to make better predictions.

REFERENCES

[1] Poria, S., Cambria, E., Gelbukh, A.: Aspect extraction for opinion mining with a deep convolutional neural network. Knowledge-Based Syst. 108, 42–49 (2016). https://doi.org/10.1016/j.knosys.2016.06.009.

[2] Balazs, J.A., Velásquez, J.D.: Opinion Mining and Information Fusion: A survey. Inf. Fusion. 27, 95–110 (2016). https://doi.org/10.1016/j.inffus.2015.06.002.

[3] Zhu, C., Cheng, G., Wang, K.: Big Data Analytics for Program Popularity Prediction in Broadcast TV Industries. IEEE Access. 5, 24593–24601 (2017). https://doi.org/10.1109/ACCESS.2017.2767104.

[4] Manek, A.S., Shenoy, P.D., Mohan, M.C., Venugopal, K.R.: Aspect term extraction for sentiment analysis in large movie reviews using Gini Index feature selection method and SVM classifier. World Wide Web. 20, 135–

154 (2017). https://doi.org/10.1007/s11280-015-0381-x.

[5] Sun, S., Luo, C., Chen, J.: A review of natural language processing techniques for opinion mining systems. Inf. Fusion. 36, 10–25 (2017). https://doi.org/10.1016/j.inffus.2016.10.004.

[6] Beigi G., Hu X., Maciejewski R., Liu H. (2016) An Overview of Sentiment Analysis in Social Media and Its Applications in Disaster Relief. In: Pedrycz W., Chen SM. (eds) Sentiment Analysis and Ontology Engineering. Studies in Computational Intelligence, vol 639. Springer, Cham. https://doi.org/10.1007/978-3-319-30319-2_13.

[7] Soleymani, M., Garcia, D., Jou, B., Schuller, B., Chang, S.F., Pantic, M.: A survey of multimodal sentiment analysis. Image Vis. Comput. 65, 3–14 (2017). https://doi.org/10.1016/j.imavis.2017.08.003.

[8] Giatsoglou, M., Vozalis, M.G., Diamantaras, K., Vakali, A., Sarigiannidis, G., Chatzisavvas, K.C.: Sentiment analysis leveraging emotions and word embeddings. Expert Syst. Appl. 69, 214–224 (2017). https://doi.org/10.1016/j.eswa.2016.10.043.

[9] Chaturvedi, I., Cambria, E., Welsch, R.E., Herrera, F.: Distinguishing between facts and opinions for sentiment analysis: Survey and challenges. Inf. Fusion. 44, 65–77 (2018). https://doi.org/10.1016/j.inffus.2017.12.006.

[10] Dragoni, M., Poria, S., Cambria, E.: OntoSenticNet: A Commonsense Ontology for Sentiment Analysis. IEEE Intell. Syst. 33, 77–85 (2018). https://doi.org/10.1109/MIS.2018.033001419.

[11] Cambria, E., Gastaldo, P., Bisio, F., Zunino, R.: An ELM-based model for affective analogical reasoning. Neurocomputing. 149, 443–455 (2015). https://doi.org/10.1016/j.neucom.2014.01.064.

[12] Dehkharghani, R., Saygin, Y., Yanikoglu, B., Oflazer, K.: SentiTurkNet: a Turkish polarity lexicon for sentiment analysis. Lang. Resour. Eval. 50, 667–685 (2016). https://doi.org/10.1007/s10579-015-9307-6.

[13] Yaqub, U., Chun, S.A., Atluri, V., Vaidya, J.: Analysis of political discourse on twitter in the context of the 2016 US presidential elections. Gov. Inf. Q. 34, 613–626 (2017). https://doi.org/10.1016/j.giq.2017.11.001.

[14] Petz, G., Karpowicz, M., Fürschuß, H., Auinger, A., Stříteský, V., Holzinger, A.: Computational approaches for mining user's opinions on the Web 2.0. Inf. Process. Manag. 50, 899–908 (2014). https://doi.org/10.1016/j.ipm.2014.07.005.

[15] Chen, T., Xu, R., He, Y., Xia, Y., & Wang, X. (2016). Learning User and Product Distributed Representations Using a Sequence Model for Sentiment Analysis. IEEE Computational Intelligence Magazine, 11(3), p34–44. doi: 10.1109/MCI.2016.2572539.

[16] Vilares, D., Alonso, M.A., Gómez-Rodríguez, C.: Supervised sentiment analysis in multilingual environments. Inf. Process. Manag. 53, 595–607 (2017). https://doi.org/10.1016/j.ipm.2017.01.004.

[17] Stieglitz, S., Meske, C., Ross, B., Mirbabaie, M.: Going Back in Time to Predict the Future - The Complex Role of the Data Collection Period in Social Media Analytics. Inf. Syst. Front. 22, 395–409 (2020). https://doi.org/10.1007/s10796-018-9867-2.

[18] Nakov, P., Rosenthal, S., Kiritchenko, S., Mohammad, S.M., Kozareva, Z., Ritter, A., Stoyanov, V., Zhu, X.: Developing a successful SemEval task in sentiment analysis of Twitter and other social media texts. Lang. Resour. Eval. 50, 35–65 (2016). https://doi.org/10.1007/s10579-015-9328-1.

[19] Fan, Z.P., Che, Y.J., Chen, Z.Y.: Product sales forecasting using online reviews and historical sales data: A method combining the Bass model and sentiment analysis. J. Bus. Res. 74, 90–100 (2017). https://doi.org/10.1016/j.jbusres.2017.01.010.

[20] Bouazizi, M., Otsuki, T.: A Pattern-Based Approach for Sarcasm Detection on Twitter. IEEE Access. 4, 5477–5488 (2016). https://doi.org/10.1109/ACCESS.2016.2594194.

[21] Hai, Z., Cong, G., Chang, K., Cheng, P., Miao, C.: Analyzing sentiments in one go: A supervised joint topic modeling approach. IEEE Trans. Knowl. Data Eng. 29, 1172–1185 (2017). https://doi.org/10.1109/TKDE.2017.2669027.

[22] Lin, C., He, Y., Everson, R., Rüger, S.: Weakly supervised joint sentiment-topic detection from text. IEEE Trans. Knowl. Data Eng. 24, 1134–1145 (2012). https://doi.org/10.1109/TKDE.2011.48.

[23] Dermouche, M., Velcin, J., Khouas, L., Loudcher, S., Dermouche, M., Velcin, J., Khouas, L., Loudcher, S., Model, A.J., Dermouche, M., Velcin, J., Khouas, L., Loudcher, S.: A Joint Model for Topic-Sentiment Evolution over Time To cite this version : HAL Id : hal-01762995 A Joint Model for Topic-Sentiment Evolution over Time. (2018).

[24] Hai, Z., Chang, K., Kim, J.J., Yang, C.C.: Identifying Features in Opinion Mining via Intrinsic and Extrinsic Domain Relevance. IEEE Trans. Knowl. Data Eng. 26, 623–634 (2014). https://doi.org/10.1109/TKDE.2013.26.

[25] Tang, D., Wei, F., Qin, B., Yang, N., Liu, T., Zhou, M.: Sentiment Embeddings with Applications to Sentiment Analysis. IEEE Trans. Knowl. Data Eng. 28, 496–509 (2016). https://doi.org/10.1109/TKDE.2015.2489653.

[26] Lim, K.W., Chen, C., Buntine, W.: Twitter-Network Topic Model: A Full Bayesian Treatment for Social Network and Text Modeling. 1–6 (2016).

[27] Sultanova, N., Kozhakhmet, K., Jantayev, R., Botbayeva, A.: Stemming algorithm for kazakh language using rule-based approach. 2019 15th Int. Conf. Electron. Comput. Comput. ICECCO 2019. 3–6 (2019). https://doi.org/10.1109/ICECCO48375.2019.9043253.

[28] M. Nandathilaka, S. Ahangama and G. T. Weerasuriya, "A Rule-based Lemmatizing Approach for Sinhala Language," 2018 3rd International Conference on Information Technology Research (ICITR), Moratuwa, Sri Lanka, 2018, pp. 1-5, doi: 10.1109/ICITR.2018.8736134.

[29] Manek, A. S., Shenoy, P. D., Mohan, M. C., & R, V. K. (2016). Aspect term extraction for sentiment analysis in large movie reviews using Gini Index feature selection method and SVM classifier. World Wide Web, 20(2), 135–154. https://doi.org/10.1007/s11280-015-0381-x.

[30] Luo, F., Li, C., Cao, Z.: Affective-feature-based sentiment analysis using SVM classifier. Proc. 2016 IEEE 20th Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2016. 276–281 (2016). https://doi.org/10.1109/CSCWD.2016.7566001.

[31] Rao, P.S., Srinivas, K., Mohan, A.K.: Stock market prices prediction using random forest and extra tree regression. Int. J. Recent Technol. Eng. (IJRTE) **8**(3), 1224–1228 (2019).

[32] Bibi, R.: Sentiment A nalysis for Urdu N ews Tweets U sing Decision T ree. 2019 IEEE 17th Int. Conf. Softw. Eng. Res. Manag. Appl. 66–70 (2019).

[33] Rao P.S., Srinivas K., Mohan A.K. (2020) A Survey on Stock Market Prediction Using Machine Learning Techniques. In: Kumar A., Paprzycki M., Gunjan V. (eds) ICDSMLA 2019. Lecture Notes in Electrical Engineering, vol 601. Springer, Singapore. https://doi.org/10.1007/978-981-15-1420-3_101.

[34] Bayhaqy, A., Sfenrianto, S., Nainggolan, K., Kaburuan, E.R.: Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes. 2018 Int. Conf. Orange Technol. ICOT 2018. 1–6 (2018). https://doi.org/10.1109/ICOT.2018.8705796.

[35] Bhargava, K., Katarya, R.: An improved lexicon using logistic regression for sentiment analysis. 2017 Int. Conf. Comput. Commun. Technol. Smart Nation, IC3TSN 2017. 2017-Octob, 332–337 (2018). https://doi.org/10.1109/IC3TSN.2017.8284501.

[36] Ramadhan, W.P., Novianty, A., Setianingsih, C.: Sentiment analysis using multinomial logistic regression. ICCREC 2017 - 2017 Int. Conf. Control. Electron. Renew. Energy, Commun. Proc. 2017-Janua, 46–49 (2017). https://doi.org/10.1109/ICCEREC.2017.8226700.

[37] Al Amrani, Y., Lazaar, M., & El Kadiri, K. E. (2018). Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis. Procedia Computer Science, 127, 511–520.

[38] I. Lakhsmi Manikyamba, Dr. A. Krishna Mohan. Novel Adaptive Multi Label Learning Classification to predict political lenience over online social networks, Adv. Res. Dyn. Control Syst. 12, 1467–1476 (2020). DOI: 10.5373/JARDCS/V12I2/S20201344.

[39] Singh, S.N., Sarraf, T.: Sentiment analysis of a product based on user reviews using random forests algorithm. Proc. Conflu. 2020 - 10th Int. Conf. Cloud Comput. Data Sci. Eng. 112–116 (2020). https://doi.org/10.1109/Confluence47617.2020.9058128.

[40] Ranjit, S., Shrestha, S., Subedi, S., Shakya, S.: Foreign Rate Exchange Prediction Using Neural Network and Sentiment Analysis. Proc. - IEEE 2018 Int. Conf. Adv. Comput. Commun. Control Networking, ICACCCN 2018. 1173–1177 (2018). https://doi.org/10.1109/ICACCCN.2018.8748819.

[41] De S Santos, R.L., De Sousa, R.F., Rabelo, R.A.L., Moura, R.S.: An experimental study based on Fuzzy Systems and Artificial Neural Netwo rks to estimate the importance of reviews about product and services. Proc. Int. Jt. Conf. Neural Networks. 2016-Octob, 647–653 (2016).

[42] Polamuri, S.R., Srinivas, K. & Mohan, A.K. Multi model-Based Hybrid Prediction Algorithm (MM-HPA) for Stock Market Prices Prediction Framework (SMPPF). Arab J Sci Eng (2020). https://doi.org/10.1007/s13369-020-04782-2.

[43] Manikyamba, I.L., Mohan, A.K.: Analyzing Political Trending Tweets for Opinion Extraction. Int. J. Recent Technol. Eng. 8, 2746–2752 (2020). https://doi.org/10.35940/ijrte.f8197.038620.