Impact of Effective Word Vectors on Deep Learning Based Subjective Classification of Online Reviews

[1]Priya Kamath B, [2]Geetha M, [3]Dinesh Acharya U, [4]Ritika Nandi and [5]Siddhaling Urolagin

[1,2,3,4]Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka, India.

[5]Department of Computer Science, Birla Institute of Technology & Science, Pilani, Dubai International Academic City, Dubai, UAE.

[1]priya.kamath@manipal.edu, [2]geetha.maiya@manipal.edu, [3]dinesh.acharya@manipal.edu, [4]ritika.nandi@learner.manipal.edu, [5]siddhaling@dubai.bits-pilani.ac.in

Correspondence should be addressed to Priya Kamath B: priya.kamath@manipal.edu

Article Info
Journal of Machine and Computing (http://anapub.co.ke/journals/jmc/jmc.html)
Doi: https://doi.org/10.53759/7669/jmc202404069
Received 10 November 2023; Revised from 27 March 2024; Accepted 18 June 2024
Available online 05 July 2024.

Abstract - Sentiment Analysis tasks are made considerably simpler by extracting subjective statements from online reviews, thereby reducing the overhead of the classifiers. The review dataset encompasses both subjective and objective sentences, where subjective writing expresses the author's opinions, and objective text presents factual information. Assessing the subjectivity of review statements involves categorizing them as objective or subjective. The effectiveness of word vectors plays a crucial role in this process, as they capture the semantics and contextual cues of a subjective language. This study investigates the significance of employing sophisticated word vector representations to enhance the detection of subjective reviews. Several methodologies for generating word vectors have been investigated, encompassing both conventional approaches, such as Word2Vec and Global Vectors for word representation, and recent innovations, such as like Bidirectional Encoder Representations from Transformers (BERT), ALBERT, and Embeddings from Language Models. These neural word embeddings were applied using Keras and Scikit-Learn. The analysis focuses on Cornell subjectivity review data within the restaurant domain, and metrics evaluating performance, such as accuracy, F1-score, recall, and precision, are assessed on a dataset containing subjective reviews. A wide range of conventional vector models and deep learning-based word embeddings are utilized for subjective review classification, frequently in combination with deep learning architectures like Long Short-Term Memory (LSTM). Notably, pre-trained BERT-base word embeddings exhibited exceptional accuracy of 96.4%, surpassing the performance of all other models considered in this study. It has been observed that BERT-base is expensive because of its larger structure.

Keywords – Natural Language Processing, Subjective Classification, Word Embeddings, Text Representations, Bidirectional Encoder Representations from Transformers.

I. INTRODUCTION

Online product reviews are so common in today's ever-expanding digital world that they play a vital role in influencing consumers' decision-making processes. Due to the explosive rise in e-commerce platforms and online transactions, people now primarily rely on product reviews to gain insight into other people’s real-world experiences before making purchases. Using computational methods, Natural Language Processing (NLP) enables machines to understand, process, and produce texts. NLP is essential for interpreting the complex and subjective components of customer feedback, as it is conveyed in textual form in online product reviews [1]. Online reviews by users generally consist of subjective and objective sentences [2]. Subjective review statements express opinions, emotions, or personal viewpoints, whereas objective review statements are more factual and present information without personal bias. Extracting subjective reviews during Sentiment Analysis is crucial for gaining deeper insight into the attitudes, preferences, and opinions of individuals, which can be highly beneficial for businesses and organizations. In addition to helping with product refinement, this procedure helps companies and customers build more transparent and reliable relationships. [3-5]

The intricacy of human language and the variety of ways people convey their thoughts and experiences make subjectivity classification in online evaluations necessary. The rapid growth of the digital marketplace presents consumers with an overwhelming range of product possibilities, complicating the decision-making process. It is essential to classify subjectivity to sort through a large number of reviews, extract useful information, and distinguish between objective evaluations and subjective opinions [2].

Word embeddings represent a significant advancement in Natural Language Processing, providing a detailed portrayal of words within a continuous vector space. In contrast to traditional methods, such as one-hot encoding, which relies on sparse representations, word embeddings offer a dense, lower-dimensional representation that captures semantic connections among words. Typically acquired through unsupervised methods, such as Word2Vec [33], Global vectors (GloVe) [35], Embeddings from Language Models (ELMo), or Bidirectional Encoder Representations from Transformers (BERT) [34], leverage extensive text datasets to generate effective word representations. In this investigation, we delve into the impact of proficient word embeddings on the subjective categorization of online restaurant reviews using deep learning methodologies with the goal of improving the accuracy of sentiment analysis systems [5]. The difficulties caused by language diversity and evolution, the techniques used in NLP for Sentiment Analysis, and the effects of subjective categorization on customer choices and commercial tactics have been examined. Subjectivity Analysis is a powerful technique that allows users to mix the overall sentiment of evaluations with the necessary information.

The key contributions of this study include:

- Effective Preprocessing: The input data is cleaned using basic pre-processing methods, and the emojis present in the data are handled effectively.
- Text representation using different Word Embeddings: The subjectivity review data are represented using various word embeddings such as Word2Vec, BERT-base, ELMo, GloVe, and ALBERT-base.
- Performance Evaluation: Evidence of notable improvements in accuracy when using the suggested word embeddings is shown.

II. WORK IN THIS AREA

The n-grams model, first proposed by Lewis [3,31] is one of the most basic statistical language models that works on simple probabilities by observing the sequence of words occurring together and their relative frequencies. A 2-gram consists of a series of two words, a 3-gram consists of a series of three words, etc. **Fig. 1** illustrates a sample N-gram model. n-gram language models evaluate the likelihood of the last word in connection with the words that came before it. Because smaller text corpuses carry a higher chance of producing findings that are radically wrong owing to outlier susceptibility, this kind of model performs best with a larger corpus. Our ability to construct intelligible sentences increases with the length of the context that is utilized to train an N-gram model. The notion that language contains long-term interdependence is a crucial consideration when using n-gram models. N-gram [32] proves to be insufficient because they are rarely extended beyond 5-grams and it is not uncommon for a sentence to require 10-grams or even more.

For text categorization applications, such as Sentiment Analysis and spam filtering, where word frequencies are more significant than word sequences, the Bag-Of-Words (BoW) model [4] is employed. This approach uses a Naive Bayes classifier to classify new sentences using a set of predefined categories (bags) of sentences. The BoW model is ineffective for tasks involving meaning because it is unable to recognize the context or infer meaning from sentences. For this reason, semantics are examined in the next section.

According to Ramos et al. [5], The TF-IDF model stands out as one of the most effective approaches for leveraging weight co-occurrence matrices in language-related challenges. The TF-IDF weight is determined by multiplying the inverse document frequency by the term frequency [6]. This facilitates the assignment of weights to the other discriminating terms. This method, which is commonly used with the Bag-of-Words paradigm, works well for simple language processing applications but becomes computationally expensive when a large vocabulary is utilized, leading to larger word vectors. Inverse document frequency may indicate the amount of information a word provides in real time, that is, whether it is common or uncommon across all texts. It represents the logarithmically scaled inverse fraction of the documents. Sparse vectors are impractical for word representation because their dimensionality is quite large (equal to the vocabulary size), and most of their values are zero. Dense vectors present a more compelling alternative when employed in many ML based systems. The dense vector square measure is computationally economical because it requires fewer parameters for estimation. Word2vec is an embedding model used to extract embeddings from a Logistic Regression or Neural Network to accomplish a classification task (predicting neighboring words). There are two variants of Word2vec namely: the Skip-Gram model (which was proposed by Mikolov et al. [7] at Google) and the CBOW model. In the skip-gram variation, the target word is predicted based on the surrounding words, while in the CBOW model, the surrounding words predict the target word. These models are shown in **Fig 1.**

In a vector space of continuous words, the words with similar meanings are represented by comparable vectors called word embeddings. Word embeddings can be created using various techniques, each with its own advantages and disadvantages. Word2Vec [36] assists in efficiently capturing semantic relationships between words. Within the vector space, words with comparable meanings are depicted as similar vectors in comparison with other embeddings, training Word2Vec models are computationally efficient. However, Word2Vec finds it difficult to learn new words, as it requires words that have already been learned. Word2Vec models are unable to capture the meanings of words with respect to a given context.

An alternative method for generating word embeddings is through GloVe, which relies on word-context matrix and matrix factorization algorithms. GloVe efficiently captured co-occurrence statistics by considering the global environment of the corpus. However, training the GloVe model requires more computing power than training Word2Vec, and rare words or words with little context in the corpus may cause problems for GloVe [35,36].

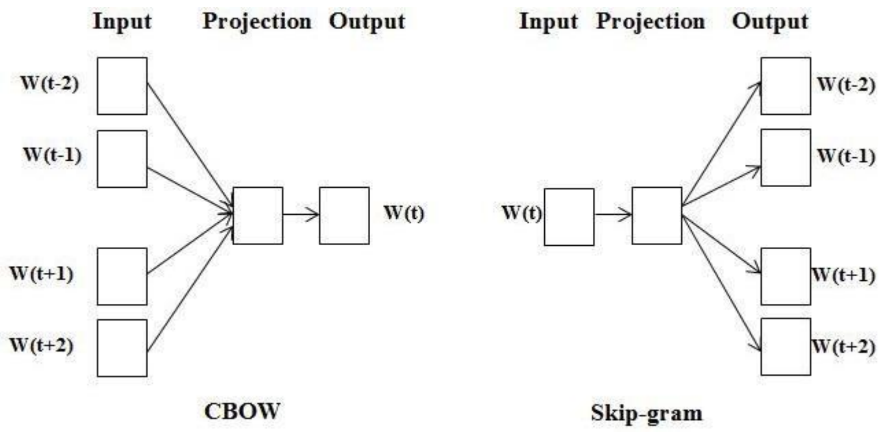


Fig. 1. CBOW and Skip-gram models

BERT utilizes a kind of deep learning-based model called transformers, where every element of output is connected to each element in the input, and the weightings between them are computed dynamically based on their connections. By capturing bidirectional context information, BERT [23] can comprehend a word's meaning with respect to the context in which it is used. BERT has been proven to be effective across a broad spectrum of tasks by attaining cutting-edge outcomes on multiple NLP benchmarks. BERT model training is computationally demanding, making it difficult for smaller businesses or projects with constrained computing resources. For applications that require fixed-size inputs, BERT[34] is less difficult to employ because it generates contextual embeddings of variable lengths.

Word2vec can even represent words that are regarded as uncommon and performs fairly well with modest quantities of training data. However, they do not completely take advantage of worldwide statistical data on word co-occurrence. By explicitly optimizing the embeddings, GloVe [8],[9] aims to establish a relationship in which the cosine similarity between two word vectors corresponds to the logarithm of the potential frequency of two words appearing adjacent to each other. Therefore, embedding vectors are needed to represent the frequency of words occurring close to them. By utilizing pre-trained vectors that are easily obtainable online, the need to develop a large vocabulary or use a skip-gram model can be avoided. This improves the performance of the model while significantly reducing computing costs. One issue with this approach is that, when working with polysemous words, the proper context cannot be recognized. After Google introduced the BERT model, numerous approaches have emerged to enhance either the computational efficiency or predictive accuracy of BERT.

RoBERTa[10,35], DistilBERT[11], XLNET[12], AlBERT[13], and XLM-RoBERTa[14] are some BERT Model variations. For the purposes of this study, only the ALBERT and BERT models were implemented in their most basic forms. “A Lite" version of the unsupervised language representation learning algorithm BERT is called ALBERT. Through the use of parameter-reducing approaches, ALBERT helps obtain the BERT-Base [15,16] model's memory constraints, and enables large-scale training.

The use of deep learning, specifically BERT models, for sentiment analysis is examined in [37,38]. The work explains how BERT models are used in sentiment analysis, detailing their performance and the improvements achieved through fine-tuning, with supporting experimental data for Stanford Sentiment Treebank dataset. The results demonstrate that BERT models are highly effective in sentiment analysis tasks.

Some of the pre-trained language models like XLNet and BERT are designed to capture the relationships between words from both directions, leading to enhanced performance in a variety of NLP tasks. This work examines the effectiveness of these two models on datasets from the “Internet Movie Database” (IMDB) consisting of 50,000 reviews, and the “Rotten Tomatoes” dataset [39]. The findings show that best results for both the datasets was achieved with XLNet model.

Existing research on word embeddings for review representation has mostly focused on how well various embedding methods capture sentiment analysis and semantic meaning. Understanding how contextual embeddings, such as BERT or contextualized word embeddings, affect the review representation of subjective datasets remains unexplored. Contextual embeddings offer dynamic, context-aware representations, whereas classic word embeddings such as Word2Vec and GloVe provide static representations. As previous studies often presume fixed-length inputs, further investigation is necessary to explore the impact of varying review lengths on word-embedding performance. [40-43] By addressing these gaps, we can better understand the advantages and limitations of various word embedding methods when it comes to representing reviews.

III. PROPOSED METHODOLOGY

Dataset Description

Cornell's subjectivity dataset [17] v1.0 contains 5000 sentences that were subjective in nature derived from Rotten Tomatoes (RT) reviews and 5000 sentences that were objective derived from IMDB plot summaries and thus making it class balanced. The data were already labeled and length-filtered. The reviews in the dataset were labelled with polarity. Statements with label '1' show that the reviews are subjective whereas the statements with polarity '0' are objective reviews. Each line in the review file corresponds to a single sentence or snippet that contains at least 10 tokens.

Data Preprocessing

The available data are in unstructured text format and hence need to be preprocessed. The review data are cleaned by eliminating various URL tags, special characters, and irrelevant symbols, as these elements do not contribute to the classification of subjective reviews. Regular expressions are used to strip out irrelevant information from the input data. To maintain dataset uniformity and prevent redundancy in word representations, all text is converted to lowercase. Applying stemming to the dataset reduces words to their root forms, which helps improve model performance. Since emojis and emoticons significantly impact sentiment detection in reviews, they are transformed into their textual equivalents.

Methodology

The performances of various word embeddings were evaluated using this publicly available dataset. The extracted features were represented in the form of a BoW [18] with unigrams, bigrams, and trigrams. The accuracy and time taken were calculated separately in each case using a Logistic Regression classifier. The overall methodology is illustrated in **Fig. 2**. The methodology proposed in this work comprises the following steps:

Step 1: Clean The Data by Eliminating Noise, Punctuation and Special Characters

Elimination of noise, punctuation symbols and special characters is considered to be one of the essential steps in the preprocessing of unstructured data. Example of noise includes any unrelated or redundant information that could lead to skewed analysis. Special characters don't serve any purpose as it does not contribute towards analyzing of sentiments from the review text. This work uses regular expressions to eliminate these noisy elements from the input data, ensuring the data is clean, consistent, and is prepared for further analysis. Elimination of noise from the input data helps in improving the accuracy and reliability of the insights derived from the data

Step 2: Split Text into Tokens That Represent Individual Words. This Process Uses Spaces as Delimiters to Separate Words Within the Input Review into Distinct Tokens

The process of splitting the text into its respective tokens is termed as Tokenization. This process generally uses whitespaces as delimiters, which means every word separated using a space in the input text becomes a separate token. For example, the sentence "I like this phone" would be split into four tokens: "I," "like," "this," and "phone." Tokenization is a crucial phase during text analysis and thus making the analysis of text easier and much organized.

Step 3: The Text Data Is Converted to Lowercase to Ensure Consistency

Conversion of all the characters present in the text to lowercase helps in ensuring uniformity across the data. This kind of data normalization helps in providing more accurate analysis. This step is performed using lower() in python. Converting the data to lower case aids in simplicity and ensures the consistency in the data. Ex: "Apple" and "apple" will be treated equally.

Step 4: Generate The Contextualized Word Embeddings

The use of pre-trained models like ELMo, ALBERT-base, and BERT generates contextualized word embeddings directly from the text data. These embeddings capture rich semantic information by considering the context of words, making them highly effective for various natural language processing tasks. For BERT, hyperparameters are fine-tuned to optimize performance, adapting the model to the specific characteristics of the dataset. The number of training epochs, which is the number of times the model processes the entire dataset, is set at 10 to balance training time and model performance. Extracting these contextualized embeddings allows the models to leverage extensive pre-existing linguistic knowledge, enhancing their ability to understand and interpret complex language patterns. This approach significantly improves the accuracy and robustness of tasks such as text classification, sentiment analysis, and named entity recognition. Additionally, fine-tuning BERT's hyperparameters ensures the model is tailored to specific project requirements, maximizing its effectiveness.

Step 5: Load Pre-Trained Word Embeddings for Word2Vec and Glove, And Extract the Fixed Length Vectors for Each Word

Loading pre-trained word embeddings for Word2Vec and GloVe involves accessing precomputed vector representations of words, capturing semantic relationships from large corpora. These embeddings provide fixed-length vectors for each word, ensuring consistent size and facilitating efficient text processing. Utilizing these vectors allows for dense, meaningful word representations that preserve semantic similarity, enhancing the performance of various natural language processing tasks. This step improves the model's ability to understand word meanings and relationships without needing extensive computational resources for training.

Step 6: *Create an architecture for a classification model and train the model.*
Designing the architecture for a classification model involves arranging the neural network, which includes determining the number and types of layers, activation functions, and interconnections between different neurons. The types of layers used in the architecture is shown in **Fig. 3**. Following the definition of the architecture, the model undergoes training using a designated dataset, where it learns to associate input data with corresponding output labels. In this work, 80% of the data is used for training and 20% of the data has been utilized for testing. Throughout the training phase, the model's parameters, encompassing weights and biases, are iteratively fine-tuned employing optimization techniques such as gradient descent. Subsequently, the model's performance is assessed using a distinct validation dataset, guiding the adjustment of its parameters to enhance accuracy and generalization. This iterative refinement process persists until the model achieves satisfactory performance on the validation set.

Step 7**:** *Assess the trained models utilizing suitable metrics namely F1-score, accuracy, precision, and recall on the test set and compare the performance of models using variety of word vectorization techniques***.**
Once the models are trained, it's vital to assess their performance using pertinent metrics such as F1-score, accuracy, precision, and recall on an independent test set.
Pseudocode for the proposed methodology is given below in **Algorithm 1.**

Algorithm 1: **Effective_Feature_Generation**

```
Input: Set of unstructured raw reviews R
Output: Preprocessed reviews effectively represented by fine tuning the word embedding models.

FUNC rem_sp_char(text):
   cleaned text ← re.sub(r'[^\w\s]', '', text)
   RETRN cleand_text

FUNC make_tokens(text):
   toks ← nltk.word_tokenize(text)
   RETRN tokens

FUNC discard_punc(toks):
   processed_tokns ← [for every token in toks if it is not in string. punc]
   RETRN processed_tokns

FUNCTION discard_stop_word(tokens):
   stop_words ← nltk. corpus.stopwords.words("english")
   processed_tokns ← [for every token in toks if it is not in list of stopwords]
   RETRN processed_tokns

FUNCTION generate_contextualized_embeddings(text_data, model):
   embeddings ← [ ]
   FOREACH sentence IN text_data:
      embeddings.append(model.generate_embeddings(sentence))
   RETURN embeddings

model ← PreTrainedModel("modelX")
model← finetune(modelX)
model_embeddings = generate_contextualized_embeddings (text_data, modelX)
FUNCTION generate_fixed_len_vectors(text_data, word_embeddings_model):
   fixed_len_vectors ← [ ]
   FOREACH word IN text_data:
      fixed_len_vectors. append(word_embeddings_model.generate_vector(word))
   RETURN fixed_len_vectors
```

These metrics offer insights into diverse facets of model effectiveness: accuracy gauges overall prediction correctness, precision quantifies the ratio of true positive predictions among all positive predictions, and recall measures the ratio of true positives correctly identified. Additionally, the F1-score strikes a balance between precision and recall, furnishing a unified metric for evaluating model performance. Moreover, it's critical to compare model performance across different word vectorization techniques. Approaches like Word2Vec, GloVe, and BERT embeddings can profoundly influence model efficacy by capturing distinct facets of word meaning and context. By contrasting models trained with varied vectorization techniques, we can pinpoint the optimal method suited to the specific task and dataset.

The work in this study, considers a basic deep-learning model. The architecture of the model including the type of layers used along with the input/output dimensions of each layer used in traditional neural embeddings such as Word2Vec and GloVE is shown in **Fig.3.** The output from the embedding layer features a third dimension, determined by the size of the word embedding. Since BERT's embedding process varies from that of other neural models, the deep learning architectures used for BERT and ALBERT embeddings are distinct. The deep learning layers used in this context include the input layer, embedding layer, LSTM layer, dropout layer, and fully connected layer.

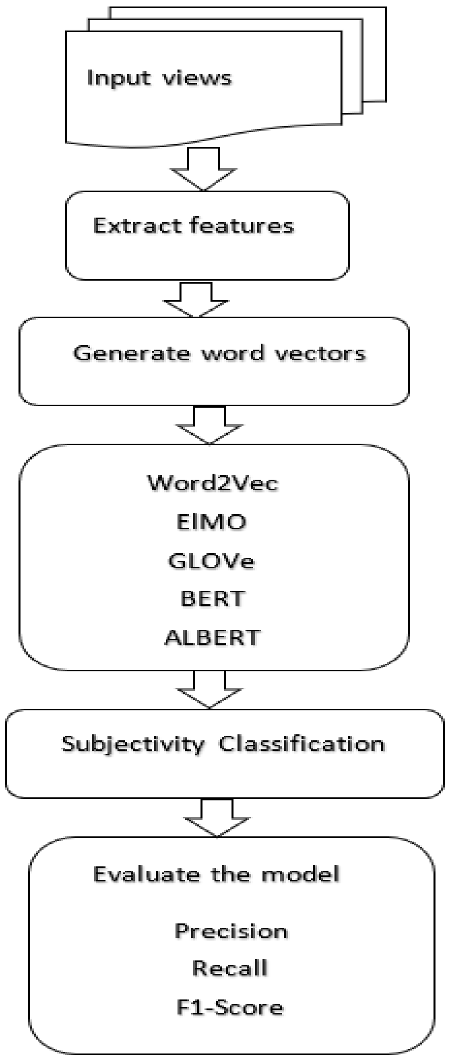


Fig 2. Overall Proposed Methodology.

Input Layer

Considered to be the initial layer of the Neural Network and is different from all other layers as it is composed of "passive" neurons. It is composed of artificial input neurons that input the first data into the system so that it can be processed further by artificial neurons in later layers.

Embedding Layer

This layer demonstrates how to convert a category variable into a reduced-dimension vector. The embedding layer in Keras was used to transfer a higher-dimensional input to a lower-dimensional vector space. As a result, it is possible to reduce the dimensionality of the data and improve visualization capabilities.

LSTM Layer

This short-term memory aids in text analysis and keeps track of the context because its architecture allows users with provisions to overlook information that is not necessary while storing keywords that may prove to be helpful later in the text. This helps us significantly extend the length of our dependencies.

Dropout Layer

This layer aids in preventing overfitting by roughly setting a portion of the input unit to zero during every update in the training process.

Fully Connected Layer

The process flattens the output from the preceding layers, converting it into a singular vector that can be utilized as input for subsequent stages.

Activation Function

Activation functions are employed to introduce non linearities into the neuron outputs and determine their activation status. Given that deep learning models aim to mimic real-world data, often intricate and nonlinear, activation functions are essential components of deep learning architectures, enabling nonlinear and complex mappings between inputs and outputs. Because deep learning relies on backpropagation, the activation functions must be differentiable. The most prevalent activation functions are sigmoid, tanh, and ReLU, which play crucial roles in facilitating the learning process.

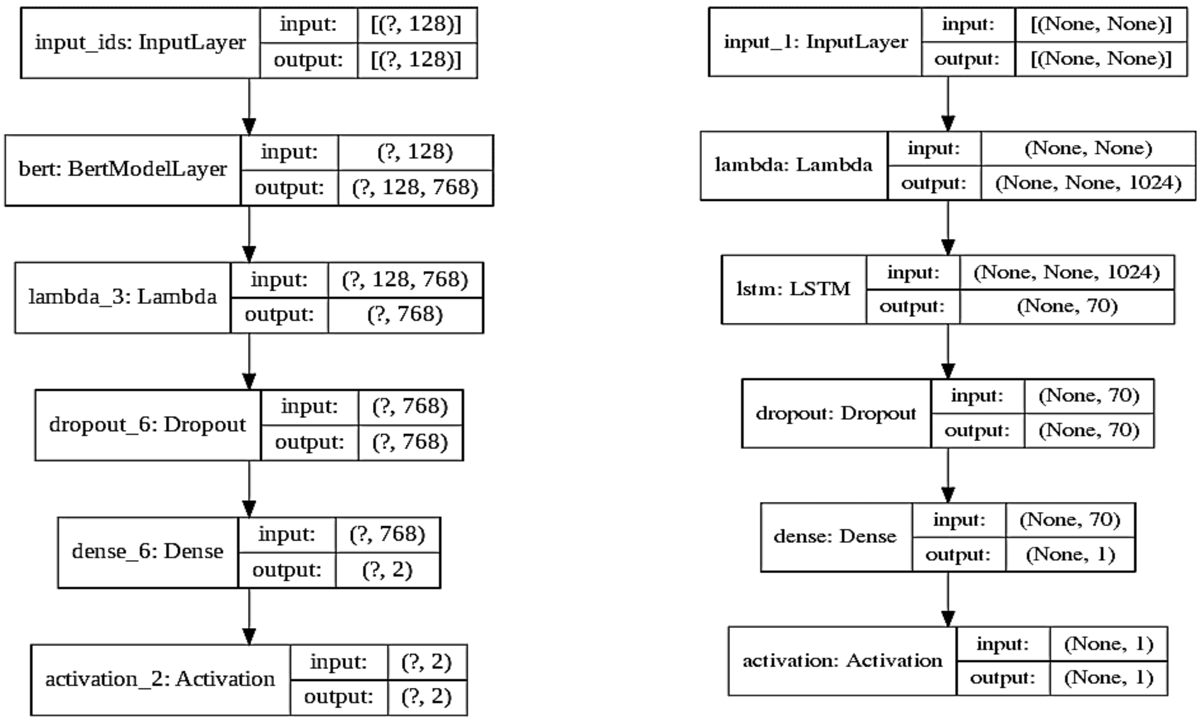

input_ids: InputLayer | input: [(?, 128)] | output: [(?, 128)]
bert: BertModelLayer | input: (?, 128) | output: (?, 128, 768)
lambda_3: Lambda | input: (?, 128, 768) | output: (?, 768)
dropout_6: Dropout | input: (?, 768) | output: (?, 768)
dense_6: Dense | input: (?, 768) | output: (?, 2)
activation_2: Activation | input: (?, 2) | output: (?, 2)
input_1: InputLayer | input: [(None, None)] | output: [(None, None)]
lambda: Lambda | input: (None, None) | output: (None, None, 1024)
lstm: LSTM | input: (None, None, 1024) | output: (None, 70)
dropout: Dropout | input: (None, 70) | output: (None, 70)
dense: Dense | input: (None, 70) | output: (None, 1)
activation: Activation | input: (None, 1) | output: (None, 1)

Fig 3. Deep Learning Model used for GloVe/Word2Vec and ELMo.

Experimental Setup

The experimental setup for the proposed model is explicitly depicted below, encompassing details such as the software and hardware utilized, and the amount of data used for training and testing.

Hardware Component

- A multi-core processor (such as an Intel Core i5 or similar)
- RAM- 8 to 16 GB.

Software Component

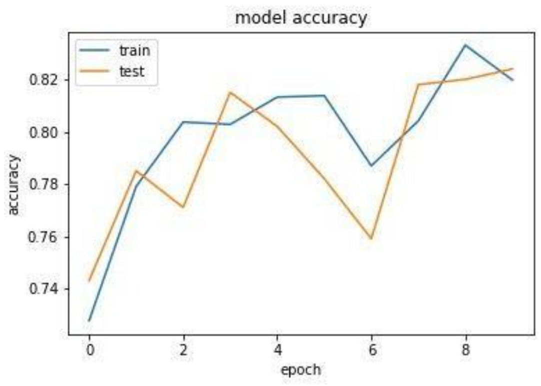
- NLTK, Gensim, and Scikit-Learn for preprocessing
- TextBlob and Machine Learning frameworks in Python
- Word Embedding models for capturing the semantic information
- Jupiter Notebook - Creating and testing analysis models

This study compares the accuracy and speed of standard word representation models with advanced developments in word embedding. While assessing the word-embedding models on the Cornell Subjectivity dataset, various state-of-the-art models were employed to assess their effectiveness in capturing subjectivity patterns within the dataset. The utilized models encompass a diverse range of techniques, spanning from traditional approaches, such as Word2Vec and GloVe, to more sophisticated contextual embeddings, such as BERT, ELMo [19], and ALBERT-Base. Each model was trained for 10 epochs, with the observation that increasing the epoch count often led to overfitting of training data.

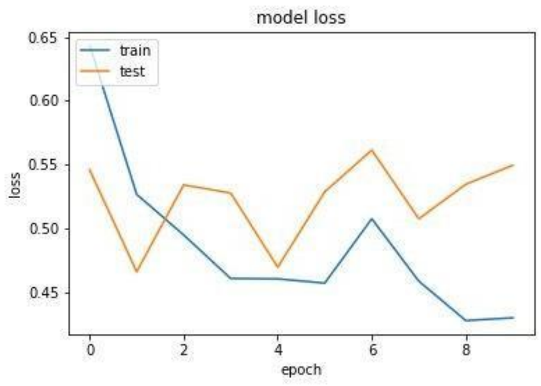
IV. EXPERIMENTAL ANALYSIS &RESULTS

Results of Word2vec

Word2Vec embeddings [20], were pre-trained on approximately hundred billion words from the Google News dataset. Three million words and phrases were represented by 300-dimensional vectors in the model. These terms were acquired by applying a straightforward data-driven methodology. The accuracy and loss of the test set are shown in **Fig 4(a)** and **Fig 4(b)** respectively.



(a)



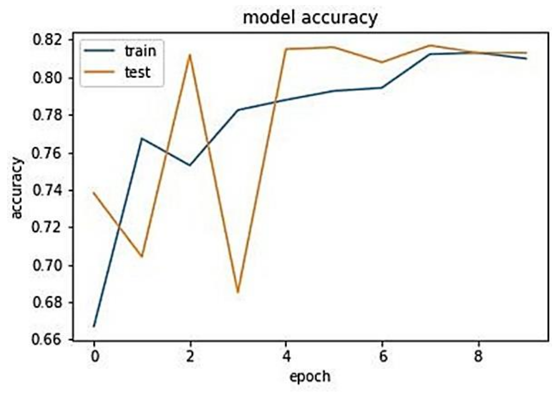
(b)

Fig 4. Word2Vec 300-dim word embeddings- (a) Accuracy (b) Loss.

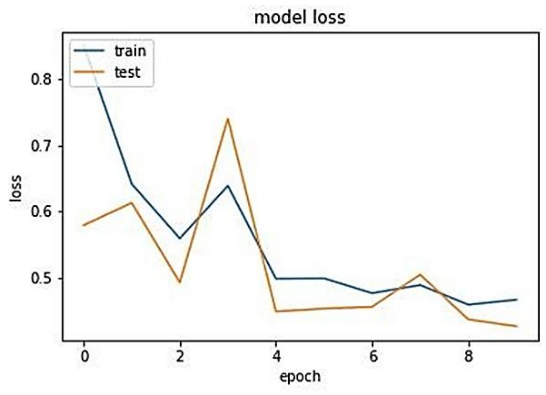
With the exception of a minor spike around the sixth epoch, the model worked very well for both the training and test data. The accuracy after ten epochs was 79.2% with an F1 score of 0.82. The reason behind This is because Word2Vec generates fixed size word embeddings.

Implementation of Glove

These embeddings [21], pre-Trained on Wikipedia 2014 + Gigaword 5 (6 B tokens, 400 K vocab, uncased, 50d, 100d, 200d, & 300d vectors), were used. The model contains 50d, 100d, 200d and 300d vectors. In this study, 300-dimensional vectors are used. The training and testing accuracy and loss over a period of 10 epochs using GloVe 300-dim word embeddings are shown in **Fig. 5(a)** and **Fig. 5(b)** respectively.



(a)



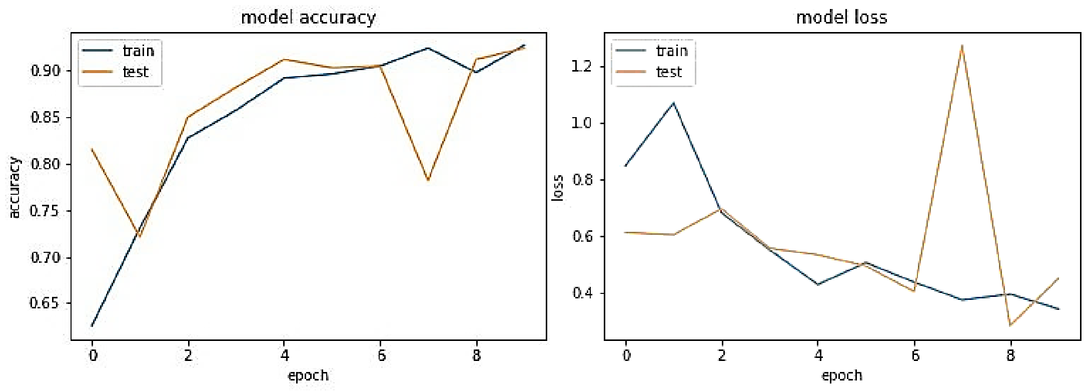
(b)

Fig 5. Glove word embeddings (a) Accuracy (b) Loss.

The performance of the model with both test and training data closely resembled that of Word2Vec. This similarity arises from the fact that both models endeavor to capture semantic connections among words by analyzing their co-occurrence patterns in extensive corpora, resulting in comparable representations for words with similar meaning. Large corpora are usually used to train both models, ensuring that they accurately reflect the variety of linguistic patterns and semantic correlations seen in natural language. The large volume of training data aids in the convergence of the models towards similar word representations. After ten epochs, GloVe Embeddings yielded an F1 score of 0.81 and an accuracy value of 81.5%.

Implementation of ELMo

ELMo embeddings are pre-trained on billions of data (approximately 800M tokens of news crawl data from WMT). In contrast to conventional static embeddings, ELMo records word meanings according to their sentence-specific context. ELMo embeddings, which are trained on a bidirectional language model, consider words that come before and after, improving their comprehension of subtleties and changes in sentiment. ELMo's contextual embeddings are useful in review analysis, because they capture the dynamic nature of sentiments and provide words with a nuanced representation that can adjust to the many emotional tones seen in reviews. The accuracy of the test data is shown in **Fig 6(a),** and the loss of test data is shown in **Fig 6(b)** respectively.



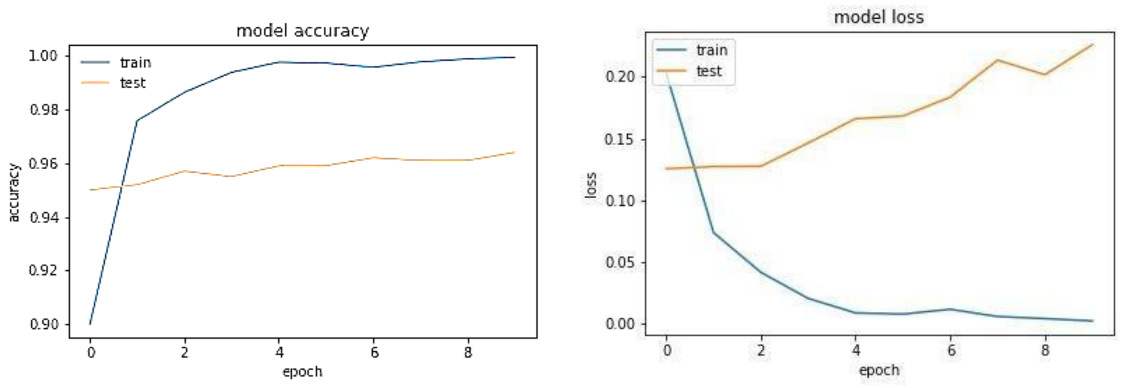
(a) (b)

Fig 6. ELMo Word Embeddings (A) Accuracy (B) Loss.

Compared with the Word2Vec and GloVe embeddings, the model demonstrated exceptional performance on both the training and test datasets. ELMo's extensive pre-training on large language corpora facilitates efficient transfer of learning to subsequent tasks. This advantage is particularly valuable when dealing with sparsely labeled data for specific tasks, as the model has already assimilated rich language features during pre-training. Despite Word2Vec and GloVe being pre-trained, their static embeddings may limit their effectiveness in knowledge transfer. Notably, the model achieved an accuracy of 91.7% and an F1 score of 0.91 after 10 epochs.

Implementation of BERT

During the implementation of BERT, the pre-training of the model and fine tuning of the BERT parameters are considered to be two crucial phases. Pretraining BERT is a very expensive operation that takes four days on four to sixteen cloud TPUs. However, this was a one-time task. Pretrained BERT embeddings based on a large text corpus (such as Wikipedia) are publicly available from Google. However, fine-tuning costs nothing and takes a few hours on a cloud GPU. Uncased embeddings of BERT, consisting of 12 layers, were used in this study. The accuracy and loss of the test set are displayed in **Fig 7(a)** and **Fig 7(b**), respectively.



(a) (b)

Fig 7. BERT word embeddings (a) Accuracy (b) Loss.

The model performs exceptionally well when compared to Word2Vec/GloVe and ELMo embeddings on the training and test data. An accuracy of 96.4% was achieved with 10 epochs. BERT captures contextual information bidirectionally, considering both the preceding and succeeding words in a sentence. As movie reviews frequently rely on the entire context and choice of words, BERT's contextual understanding helps produce a better idea of sentiment. BERT can handle the change in tone and style of movie reviews because of its pre-training exposure to a variety of language styles. These factors collectively position BERT as a powerful embedding model for analyzing subjective sentences from movie reviews.

Implementation of ALBERT

For the ALBERT [23,24] model, BOOK CORPUS and English Wikipedia are used for pretraining the baseline models, such as the BERT model. Similar to BERT, a vocabulary size of 30,000 was tokenized using Sentence Piece. The accuracy and loss on the test set were shown in **Fig 8(a)** and **Fig 8(b)** respectively.

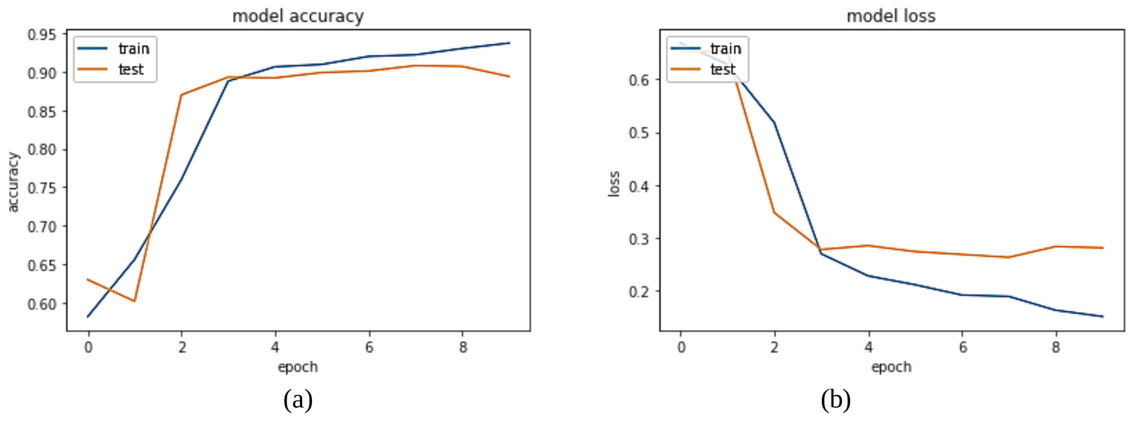


Fig 8. ALBERT word embeddings (a) Accuracy (b) Loss.

The model performs exceptionally when compared to Word2Vec/ GloVe and however, it does not perform as well as the BERT embeddings on the training and test data. The accuracy obtained after 10 epochs was 91.2% with an F1 score of 91%. The performances of various word vector representations and embeddings are tabulated in **Table 1** and plotted in **Fig 9**. The results obtained were compared based on accuracy, weighted average of recall, F1-score, and precision.

Table 1. Performance of Various Text Representation Methods

Word Embeddings	Accuracy	Precision	Recall	F1-score
Word2Vec	79.2	81.5	81.5	81.5
GloVe	81.5	82	83	82
ELMo	91.3	91	91	91
BERT-Base	96.4	96	96	96
ALBERT-Base	91.2	91	91	91

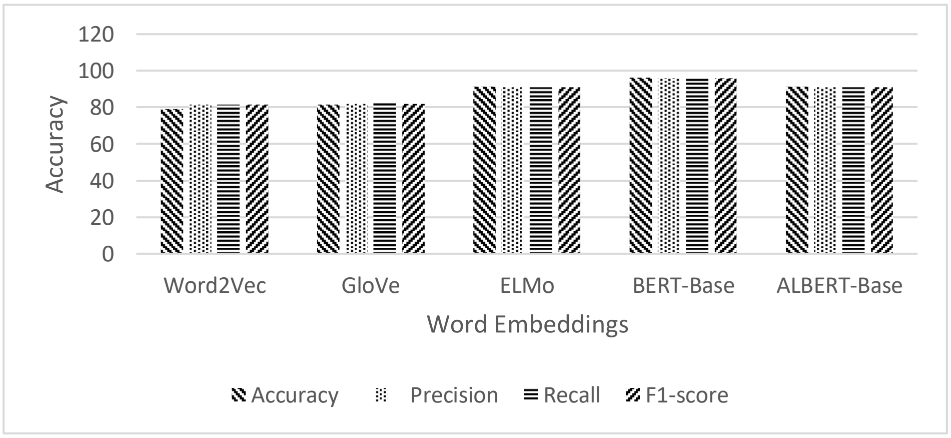


Fig 9. Comparison Of Word Embedding Performance.

The findings indicate that the BERT model significantly outperforms all other word-embedding methods. The BERT-base is computationally more expensive because of its larger structure, even though it performs noticeably better than GloVe, Word2Vec, and ELMo and has more parameters [25]. Dataset coverage [17] is biased towards the restaurant domain, potentially limiting its representation across diverse contexts. The subjective/objective labels assigned to the review statements lack consistency because of the variations in annotator interpretations, and the size of the dataset is comparatively smaller, thereby hindering the performance of the models trained on it, especially for tasks requiring large amounts of training data. The temporal bias of the dataset leads to outdated representations of language use and sentiment, diminishing its relevance in the current context.

Taken together, the findings indicate that the field of subjective review analysis can be greatly advanced by utilizing sophisticated word embeddings, such as BERT-base. This can result in enhanced applications across a range of domains, such as e-commerce, customer feedback analysis, personalized recommendations, and improved accuracy. The comparative analysis conducted between the proposed methodology and the established benchmark results reveals that employing a BERT-base for classifying subjective reviews has demonstrated remarkable efficiency, achieving a notable accuracy exceeding 95%. This improvement was substantiated by comparative studies [26-31] which consistently demonstrated enhanced performance when integrating this approach. [32-35]

V. CONCLUSION

In this work, the performance of various word embeddings including Word2Vec, BERT, GloVe, ELMo, and ALBERT-Base, was investigated on the Cornell Subjective Review dataset for subjective classification. The objective is to evaluate the effectiveness of these embeddings in capturing nuanced sentiments and subjective information from online reviews. Through rigorous experimentation and analysis, it was observed that BERT-Base consistently outperformed other word embeddings in terms of accuracy for subjective classification on the Cornell dataset. [36-43] Although Word2Vec, GloVe, ELMo, and ALBERT-Base performed well, they were unable to fully capture the many details of subjective language seen in the Cornell Subjective Review dataset because of their static or less context-aware nature. Even though Word2Vec and GloVe have demonstrated success in capturing semantic links, their limited adaptability and lack of contextual information may present challenges when working with subjective language. ALBERT-Base, which was designed to be more computationally efficient than BERT, did not outperform BERT-Base in this specific task, highlighting the significance of BERT's architecture in capturing subjective terms.

In future, methods for capturing complex semantic relationships will be studied and investigated for domain-specific knowledge. Traditional word embeddings can be combined with context-based word-embedding models that leverage transformer architecture. Moreover, a hybrid model with these word embeddings can be developed to effectively represent reviews in vector format. Additionally, transfer-learning-based methods and Large Language Models will be explored for pre-trained embeddings across diverse domains and tasks.

Author Contributions

All authors have contributed equally.

Funding

This research has not received any kind of external funding.

Ethical statement

All the authors mentioned in the manuscript have agreed for authorship, read, approved and given consent for submission.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1]. M. Arslan and C. Cruz, "Leveraging NLP approaches to define and implement text relevance hierarchy framework for business news classification," Procedia Computer Science, vol. 225, pp. 317–326, 2023, doi: 10.1016/j.procs.2023.10.016.

[2]. D. Jannach, "Evaluating conversational recommender systems," Artificial Intelligence Review, vol. 56, no. 3, pp. 2365–2400, Jul. 2022, doi: 10.1007/s10462-022-10229-x.

[3]. Cavnar, William B., and John M. Trenkle. "N-gram-based text categorization." Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval. Vol. 161175. 1994.

[4]. Sarkar, Atanu, Anil Bikash Chowdhury, and Mauparna Nandan. "Classification of Online Fake News Using N-Gram Approach and Machine Learning Techniques." Doctoral Symposium on Human Centered Computing. Singapore: Springer Nature Singapore, 2023.

[5]. Das, Mamata, and P. J. A. Alphonse. "A comparative study on tf-idf feature weighting method and its analysis using unstructured dataset." arXiv preprint arXiv:2308.04037 (2023).

[6]. T. Hasan and A. Matin, "Extract Sentiment from Customer Reviews: A Better Approach of TF-IDF and BOW-Based Text Classification Using N-Gram Technique," Proceedings of International Joint Conference on Advances in Computational Intelligence, pp. 231–244, 2021, doi: 10.1007/978-981-16-0586-4_19.

[7]. Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems 26 (2013).

[8]. Dharma, Eddy Muntina, et al. "The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification." J Theor Appl Inf Technol 100.2 (2022): 31.
[9]. W. K. Sari, D. P. Rini, and R. F. Malik, "Text Classification Using Long Short-Term Memory With GloVe Features," Jurnal Ilmiah Teknik Elektro Komputer dan Informatika, vol. 5, no. 2, p. 85, Feb. 2020, doi: 10.26555/jiteki.v5i2.15021.
[10]. Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).
[11]. Sanh, Victor, et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." arXiv preprint arXiv:1910.01108 (2019).
[12]. Yang, Zhilin, et al. "Xlnet: Generalized autoregressive pretraining for language understanding." Advances in neural information processing systems 32 (2019).
[13]. Wang, Hanqi, Xiaoli Hu, and Huibing Zhang. "Sentiment analysis of commodity reviews based on ALBERT-LSTM." Journal of Physics: Conference Series. Vol. 1651. No. 1. IOP Publishing, 2020.
[14]. Xie, Shuyi, et al. "PALI at SemEval-2021 task 2: fine-tune XLM-RoBERTa for word in context disambiguation." arXiv preprint arXiv:2104.10375 (2021).
[15]. M. P. Geetha and D. Karthika Renuka, "Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model," International Journal of Intelligent Networks, vol. 2, pp. 64–69, 2021, doi: 10.1016/j.ijin.2021.06.005.
[16]. Xu, Hu, et al. "BERT post-training for review reading comprehension and aspect-based sentiment analysis." arXiv preprint arXiv:1904.02232 (2019).
[17]. Cornell Subjectivity Dataset: "Movie Review Data". https://www.cs.cornell.edu/people/pabo/movie-review-data/
[18]. W. A. Qader, M. M. Ameen, and B. I. Ahmed, "An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges," 2019 International Engineering Conference (IEC), Jun. 2019, doi: 10.1109/iec47844.2019.8950616.
[19]. K. Ethayarajh, "How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings," Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, doi: 10.18653/v1/d19-1006.
[20]. M. Grohe, "word2vec, node2vec, graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data," Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Jun. 2020, doi: 10.1145/3375395.3387641.
[21]. J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, doi: 10.3115/v1/d14-1162.
[22]. A. van Loon and J. Freese, "Word Embeddings Reveal How Fundamental Sentiments Structure Natural Language," American Behavioral Scientist, vol. 67, no. 2, pp. 175–200, Feb. 2022, doi: 10.1177/00027642211066046.
[23]. Y. Liu, Z. Yin, C. Ni, C. Yan, Z. Wan, and B. Malin, "Examining Rural and Urban Sentiment Difference in COVID-19–Related Topics on Twitter: Word Embedding–Based Retrospective Study," Journal of Medical Internet Research, vol. 25, p. e42985, Feb. 2023, doi: 10.2196/42985.
[24]. R. Patil, S. Boit, V. Gudivada, and J. Nandigam, "A Survey of Text Representation and Embedding Techniques in NLP," IEEE Access, vol. 11, pp. 36120–36146, 2023, doi: 10.1109/access.2023.3266377.
[25]. G. S, D. T, and A. Haldorai, "A Supervised Machine Learning Model for Tool Condition Monitoring in Smart Manufacturing," Defence Science Journal, vol. 72, no. 5, pp. 712–720, Nov. 2022, doi: 10.14429/dsj.72.17533.
[26]. J. Mutinda, W. Mwangi, and G. Okeyo, "Sentiment Analysis of Text Reviews Using Lexicon-Enhanced Bert Embedding (LeBERT) Model with Convolutional Neural Network," Applied Sciences, vol. 13, no. 3, p. 1445, Jan. 2023, doi: 10.3390/app13031445.
[27]. M. Qorich and R. El Ouazzani, "Text sentiment classification of Amazon reviews using word embeddings and convolutional neural networks," The Journal of Supercomputing, vol. 79, no. 10, pp. 11029–11054, Feb. 2023, doi: 10.1007/s11227-023-05094-6.
[28]. A. Areshey and H. Mathkour, "Transfer Learning for Sentiment Classification Using Bidirectional Encoder Representations from Transformers (BERT) Model," Sensors, vol. 23, no. 11, p. 5232, May 2023, doi: 10.3390/s23115232.
[29]. Maas, Andrew, et al. "Learning word vectors for sentiment analysis." Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies. 2011.
[30]. M. Giatsoglou, M. G. Vozalis, K. Diamantaras, A. Vakali, G. Sarigiannidis, and K. Ch. Chatzisavvas, "Sentiment analysis leveraging emotions and word embeddings," Expert Systems with Applications, vol. 69, pp. 214–224, Mar. 2017, doi: 10.1016/j.eswa.2016.10.043.
[31]. Garrido-Merchan, Eduardo C., Roberto Gozalo-Brizuela, and Santiago Gonzalez-Carvajal. "Comparing BERT against traditional machine learning models in text classification." Journal of Computational and Cognitive Engineering 2.4 (2023): 352-356.
[32]. M. García, S. Maldonado, and C. Vairetti, "Efficient n-gram construction for text categorization using feature selection techniques," Intelligent Data Analysis, vol. 25, no. 3, pp. 509–525, Apr. 2021, doi: 10.3233/ida-205154.
[33]. A. Mallik and S. Kumar, "Word2Vec and LSTM based deep learning technique for context-free fake news detection," Multimedia Tools and Applications, vol. 83, no. 1, pp. 919–940, May 2023, doi: 10.1007/s11042-023-15364-3.
[34]. G. Nasreen, M. Murad Khan, M. Younus, B. Zafar, and M. Kashif Hanif, "Email spam detection by deep learning models using novel feature selection technique and BERT," Egyptian Informatics Journal, vol. 26, p. 100473, Jun. 2024, doi: 10.1016/j.eij.2024.100473.
[35]. Diaz Tiyasya Putra and Erwin Budi Setiawan, "Sentiment Analysis on Social Media with Glove Using Combination CNN and RoBERTa," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 7, no. 3, pp. 457–563, Jun. 2023, doi: 10.29207/resti.v7i3.4892.
[36]. P. Rakshit and A. Sarkar, "A supervised deep learning-based sentiment analysis by the implementation of Word2Vec and GloVe Embedding techniques," Multimedia Tools and Applications, Apr. 2024, doi: 10.1007/s11042-024-19045-7.
[37]. Y. Wu, Z. Jin, C. Shi, P. Liang, and T. Zhan, "Research on the application of deep learning-based BERT model in sentiment analysis," Applied and Computational Engineering, vol. 71, no. 1, pp. 14–20, May 2024, doi: 10.54254/2755-2721/71/2024ma.
[38]. A. Sharma and D. B. Jayagopi, "Modeling essay grading with pre-trained BERT features," Applied Intelligence, vol. 54, no. 6, pp. 4979–4993, Mar. 2024, doi: 10.1007/s10489-024-05410-4.
[39]. M. M. Danyal, S. S. Khan, M. Khan, S. Ullah, F. Mehmood, and I. Ali, "Proposing sentiment analysis model based on BERT and XLNet for movie reviews," Multimedia Tools and Applications, Jan. 2024, doi: 10.1007/s11042-024-18156-5.
[40]. S. Kumar, U. Gupta, A. K. Singh, and A. K. Singh, "Artificial Intelligence," Journal of Computers, Mechanical and Management, vol. 2, no. 3, pp. 31–42, Aug. 2023, doi: 10.57159/gadl.jcmm.2.3.23064.
[41]. N. Ranjan, "Enhancing Voting Security and Efficiency," Journal of Computers, Mechanical and Management, vol. 2, no. 3, pp. 9–15, Aug. 2023, doi: 10.57159/gadl.jcmm.2.3.23065.
[42]. S. B. Kulkarni and S. Kulkarni, "Study of the Value of π Probability Sampling by Testing Hypothesis and Experimentally," Journal of Computers, Mechanical and Management, vol. 3, no. 1, pp. 22–29, Feb. 2024, doi: 10.57159/gadl.jcmm.3.1.240101.
[43]. N. Kumar, U. Dugal, and A. Singh, "Optimizing Task Scheduling in Cloud Computing Environments using Hybrid Swarm Optimization," Journal of Computers, Mechanical and Management, vol. 2, no. 5, pp. 08–13, Oct. 2023, doi: 10.57159/gadl.jcmm.2.5.23076.