**Abstract :**

Sentiment analysis has been the focus of researchers' great attention, as it has wide applicability and tremendous utility value in multiple domains. The classification of sentiment in text is done through many approaches and techniques. This thesis will focus on tuning hyperparameters to optimize the performance of sentiment analysis using Word2Vec and deep learning models for Twitter data. Hence, it first performs preprocessing and converts the raw tweets into high-dimensional vectors using the Word2Vec Skip-gram model, which captures semantic relationships between words. The vector representations are then given as input to deep neural networks tailored for sentiment classification. Most of the emphasis in this research would be put on hyperparameters—dimensions, epochs, learning rate, window size, hidden layers, batch size, and optimizer—because these greatly influence the model performance. Through rigorous experimentation, it identifies the key hyperparameter configurations for improved sentiment classification accuracy. The results have shown that fine-tuning these parameters can significantly improve the performance of a model, hence making the analysis of large-scale social media data more efficient. This work will contribute to the advancement of natural language processing in optimizing sentiment analysis models using Word2Vec and deep learning techniques.

Keywords: Sentiment Analysis, Twitter, Word2Vec, Deep Learning, Hyperparameter Optimization, Neural Networks

**Introduction :-**

Sentiment analysis, a subfield of natural language processing, is the classification of text into categories such as positive, negative, or neutral, with the aim of identifying the underlying emotional tone. This task has gained importance in the area of social media, where there is a huge volume of user-generated content—like tweets—that can provide insight into public opinion, market trends, and societal issues. With the real-time and high-volume nature of data on Twitter, it becomes a rich source for sentiment analysis, hence a favorite for many research studies and practical applications.

The traditional methods of sentiment analysis, such as bag-of-words or TF-IDF (Term Frequency-Inverse Document Frequency), usually fail to capture the deeper semantic relationships between words. With this limitation, the Word2Vec model has been widely adopted since it was introduced by Mikolov et al. Distributed vector representations for words in a neural network—where semantically similar words are mapped to proximate points in a high-dimensional space—are learnt by Word2Vec. The Skip-gram approach within Word2Vec is particularly effective in capturing contextual word meanings, which really makes it a powerful tool for text vectorization in sentiment analysis.

In recent years, deep learning models, including feedforward neural networks and recurrent neural networks (RNNs), have been applied to the task of sentiment classification with a lot of improvement over the baseline performances. However, these models are highly sensitive toward the choice of hyperparameters like the number of hidden layers, learning rate, batch size, and optimizer used. The hyperparameter tuning step is one very crucial step while fine-tuning the performance of any model to its best possible accuracy.

This thesis tries to study the effect of hyperparameter optimization in the context of sentiment analysis for Twitter data. Using Word2Vec Skip-gram as the word embedding combined with deep learning architectures for sentiment classification, this study focuses on determining the most important hyperparameters that would yield the best performance. The study systematically goes about fine-tuning various hyperparameters, including embedding dimensions, window size, epochs, and learning rate, in order to evaluate their influence on the accuracy of sentiment prediction. In this work we implemented the skipgram model without using library inorder to understand how backpropogation is implemented in skipgram modeland also all preprocessing task like text-cleanup process, removing stopwords all done without using the inbuilt library.

Results of this research will definitely contribute to the development of more efficient and accurate sentiment analysis models, mainly by providing an outline for efficient analysis of vast social media information in real time. The current work will help push forward the boundaries of what has been possible so far in sentiment analysis using Word2Vec-based deep learning with far-reaching implications in marketing, politics, and social research.

**Related work:**

Sentiment analysis is the study of how opinions and views can be associated with emotions and attitude displays in natural language with respect to an occurrent or event. We say that the area of research that analyzes thoughts, feelings, perceptions, behaviors and emotions of people towards things like goods, products, political parties, people, problems, incidents, issues and their attributes is referred to as sentiment analysis. One researcher uses a different technique for predicting social opinion and emotion through text and languages of sentiment analysis.

[1] The paper compares the different Machine Learning approaches, such as Naïve Bayes Classification method, Support Vector Machine Classification Method, and Maximum Entropy Classification method. It shows how sentiments analysis is done by this classification algorithm and what is the accuracy and precision in these cases.

[2] The proposed research of the sentiment analysis conducted by using the TF-IDF method and the Naive Bayes Classifier based on reviews from buyers. The data collected is 1000 reviews, which are divided into 700 training data and 300 test data. The next stage is preprocessing of the text, including case folding—converting all uppercase letters to lowercase, tokenizing—separating the sentences into single words, stop words—removing tokenizing conjunctions that have nothing to do with sentiment analysis, stemming—changing words into basic word forms, and word weighting with TF-IDF. Lastly, it classifies. From its classification results, an accuracy rate of 80.2223% was obtained.

[3] This research compares the performance of TF-IDF and Word2Vec as a feature representation technique for the emotional classification of text based on data from commuter lines and TransJakarta tweets. These are followed by two classifiers in a two-class classification process using Support Vector Machine (SVM) and Multinomial Naïve Bayes (MNB). The first stage is the classification of whether a text contains emotion or not; the second stage classifies texts with emotions into five categories: happy, angry, sad, scared, and surprised. In this paper, three schemes are considered: SVM with TF-IDF, SVM with Word2Vec, and MNB with TF-IDF. This research compares the different scenarios through an analysis aimed at bringing forth the influence of the feature representation method and classification algorithms on the accuracy in emotional text classification.

[4] This paper proposes a model that works on Natural Language Processing and deep neural networks in improving text classification, especially for short sentences. It first performs detailed data cleaning, then uses a pre-trained Word2Vec model to generate word embeddings that include semantic relations among words. The embeddings are then fed into the Convolutional Neural Network (CNN) layer for further extraction of important features used in effective categorization. Combining Word2Vec as a word representation with CNN for feature extraction, the model got the capability to classify short text data more effectively, therefore improving accuracy and performance in tasks such as sentiment analysis or other text classification problems.

[5]The study suggests Twitter is used by Indonesian to express their sentiment in the form of tweets. This study used word2vec approach in extracting features by transforming data into vector. For word2vec model, this study used Wikipedia in Bahasa Indonesia as a massive amount corpus for model training. Study used support vector machine (SVM) for the classification process. Sentiment classification process was carried out through. Processing the training data in the form of tweets, and then transformed them into a model for testing using the test data. Indonesia has been chosen to use as a large amount of corpus for the training model. The training is by using Continuous Bag-of-Word (CBOW) and skip-gram as word2vec model architecture.

[6] This paper reviews recent research applying deep learning techniques to sentiment analysis, mainly on sentiment polarity detection. It considers models that use methods like Term Frequency-Inverse Document Frequency (TF-IDF) and word embeddings to process different datasets. The architectures of DNN, CNN, and RNN were analyzed and combined with word embedding and TF-IDF to carry out the task of sentiment analysis.

[7] This work tries to empirically prove that there exists an optimal set of hyperparameters for Word2Vec and measures many combinations against the original publicly released Word2Vec embeddings. Experiments include intrinsic and extrinsic evaluations such as named entity recognition and sentiment analysis. It underlines that the best-performing model is usually task-specific, where high analogy scores do not go hand in hand with the F1 scores, and performance is not a function of the data size alone. The ethical aspects related to the saving of time, energy, and resources hint that sometimes smaller corpora can perform equally well or even better than large ones. It also shows that increasing the embedding dimensions beyond a point hurts performance and using smaller corpora yields better WordSim scores, Spearman correlation, and downstream task performance compared to the original Word2Vec model trained on a 100 billion-word corpus.

**Proposed methodology**

In our research work, we mainly implemented two models. Firstly, we build a word2vec skipgram model which trains the given tweet dataset to generate word vectors for each word in the dataset without using the inbuilt libraries. Secondly, we use this word vectors to generate word embeddings and sentence embedding for each tweet, and build a deep neural network model to train the corresponding tweet with its sentiment label(positive/negative/neutral). After training, this model can be used to predict sentiments of a given tweet. This study aims to examine the impact of various parameters like Learning rate, Optimizer, Epochs, Batch Size, Number of hidden layers, Number of Neurons per Layer, Dropout rate, Weight Initialization on the accuracy and optimization of both models.

A diagram of a flowchart

Description automatically generated

a) Dataset Collection :The Sentimental Analysis dataset was collected from the Kaggle repository [7], Sentimental Analysis Dataset is a collection of data from tweets and organised in the form of dataset. This dataset has 3535 entries with 9 attributes . The 9 attributes are textid, text, sentiment, time of tweet, age of user, country, population-2020, land area, density. Textid, text give details about the tweets, sentiment tells whether the tweet is classified as positive/negative/neutral. In this dataset for sentiment analysis we need two features text and sentiments. Table 1 shows the data description of the sentiment analysis tweet dataset.

TABLE 1

b) Dataset Preprocessing :The Preprocessing stage is the the process of preparing dataset before it is processed in the system. Preprocessing is used in this study to select data so that processed data becomes more structured. Here we remove all the NaN values in the dataset. We add a new attribute in the dataset which represent numerical labels for positive/negative/neutral as 0,1,2 respectively. Then we select the most relevant features from the dataset that is needed for the sentimental analysis. In this dataset we select text as the X-Feature and Numerical labels as Y-Feature

c) Text data preprocessing: X feature contain text data with tweets which consists of unnecessary characters for the sentiment classification process such as URLs, mentions, hashtags, etc. The dataset needed to go through a text clean-up process. This process also used stopwords for removing frequent words that were less meaningful. Every tweet after clean-up will be tokenized.

d) Word2vec model: A corpus is created with all the tokens in the pre-processed tweet. Onehot encoding is made for each token. We implement skip gram model for training. Skipgram models is widely used word2vec model for word embeddings, model is trained in such a way that token(centerword) should give its context words as its output. Word2vec process is as follows- Firstly the one-hot encoding of the centerword(any token) is passed through first weight matrix and converted into N-dimensions, then passed through second weight matrix and converted to the original dimension vector. This obtained vector is compared with context vectors to calculate loss. Then backpropogation is initiated to update the weight by calculating the gradient with respect to loss and weight, subtracted from the weight matrices. In each epoch this process continues to minimize the loss. After training the model, first weight matrix is taken as the word-embedding of all tokens.

e) Sentimental analysis model – The pre-processed tweet data are tokenised. Using word2vec model we convert all the tokens into word embedding and find the average of all tokens in a tweet to form sentence vector. This sentence vector is passed through a deep neural network containing multiple hidden layers and activation functions, to form a three-dimensional vectors. Eventually this three-dimensional vector is passed through SoftMax function which gives a probability distribution as output. This output is compared with the sentiment label using CrossEntropyloss.  
Backward propagation is initiated after calculating the loss in each epoch. After training the model , we are able to predict the sentiment of a given tweet

f) Evaluation – Model’s accuracy and loss is evaluated on the basis of change in parameters like Learning rate, Optimizer, Epochs, Batch Size, Number of hidden layers, Number of Neurons per Layer, Dropout rate, Weight Initialization.