# A comparison of machine learning and deep learning techniques for textual feedback analysis

Urooj Abdel Halim and Syed Zaffar Qasim

**Abstract – The higher educational institutions gather student feedback after the end of each semester to improve the quality of education. The feedback consists of a grading scale to answer the questions followed by a textual response conveying the sentiments regarding the student's experience. Since there is a considerable amount of response influx, going through every single textual feedback is time-consuming; hence the need arises to extract sentiments from individual comments and classify them as positive, negative, or neutral. Our research aimed to compare different machine learning and deep learning techniques for developing an effective sentiment classification system for instructors. In this study, we analyzed student feedback consisting of 19000 comments and trained various machine learning and deep learning algorithms using several feature extraction techniques. Among the different algorithms employed, a cascading neural network consisting of CNN combined with LSTM using Glove word embedding outperformed all the other architecture giving an accuracy of 91.27 %.**

**Keywords** Sentiment Analysis, Opinion mining, Text mining, Feedback Analysis, Deep learning

## 1 Introduction

Sentiment analysis involves analyzing the textual appraisal and opinions of people toward objects and their properties [1,2]. The objects include products, services, persons, organizations, topics, etc. It is also called opin- ion mining and has gained significant attention in re- cent years as a powerful tool for understanding and an- alyzing human emotions, opinions, and attitudes from various types of texts. One of the most critical areas where sentiment analysis can be applied is in the field of education, specifically in analyzing student feedback on instructors. A fundamental step in this analysis is to find the sentiment orientation, also called polarity, of some textual unit e.g., a single comment of a student. This polarity can be categorized as fixed labels which are positive, negative and neutral.

This research paper aims to apply sentiment analysis techniques to student feedback data collected from the RateMyProfessors.com website, a popular platform where students can rate and provide feedback on their instructors and academic environment. The site contains a wealth of information, including student evaluations of instructors, textual comments, and overall ratings. The data from this website can provide valuable insights into student perceptions of instructors, which can be used to improve the quality of education and enhance the learning experience. A distinctive feature of textual feedback is that it allows the students to identify various issues and problems that are otherwise not possible with scale-based scores. Additionally, the students can give valuable suggestions for improvement in course management and syllabus modification.

In this research, two different approaches have been used for sentimental analysis of student feedback. The first approach is a deep learning approach, which is like the one proposed by Aytuğ Onan [3]. Onan's approach uses a deep learning model to analyze student feedback and extract opinions and sentiments from the text. The second approach that is explores machine learning algorithms with a feature extraction technique, like the one proposed by Tamrakar *et. al.* [4]. This approach uses various feature extraction techniques to extract relevant information from the student feedback data and classify it into different sentiments.

The goal of this research is to compare the performance of these two approaches and determine which is more effective for sentiment analysis of student feedback on instructors. We hope that the results of this study will provide valuable insights into student

perceptions of instructors and help the education community to improve the quality of education.
The remaining part of this paper is organized as follows. In section 2, we present a review of past work related to sentiment analysis of textual feedback using machine learning and deep learning techniques. In the next section, 3, the detailed methodology for carrying out the textual feedback analysis is presented. It includes the description of the dataset, proposed framework, feature extraction, and the different machine learning and deep learning techniques. Then section 4 covers the details of the experiments conducted, and their results, and a discussion on the analysis. The conclusion of this study will be covered in section 5.

## 2 Related work

### 2.1 Machine Learning Techniques for Sentimental Analysis

The authors in [5] used a lexicon-based approach to classify student comments as positive, negative, or neutral. They used a dataset of student comments and applied a lexicon-based approach, which utilizes a lexicon of words and their corresponding sentiment scores. The sentiment scores were calculated based on the number of positive and negative words in a comment. The authors compared the performance of the lexicon-based approach with other machine learning algorithms such as Naive Bayes and Decision Tree. They found that their lexicon-based approach performed better in terms of accuracy and F1-score.
This paper [6] presents a study of using machine learning to predict learning-related emotions from students' textual classroom feedback via Twitter. The authors collected a dataset of students' feedback from Twitter and used machine learning algorithms such as Naive Bayes, Decision Tree, and Random Forest to clas- sify the feedback into different emotion categories such as frustration, confusion, satisfaction, and engagement. They also used pre-processing techniques to clean the data, such as removing stop words, stemming, and feature selection. The results showed that the Random Forest algorithm performed the best among the algorithms, with an accuracy of 70.2%.
Nasim et al. [7] proposed a hybrid approach using machine learning and lexicon-based methods for sentiment analysis of students' feedback. The two machine learning algorithms used were Random Forest and Support Vector Machines. The sentiment analysis model was trained using a combination of TF-IDF and lexicon-based features. A comparative analysis was carried out between the proposed model and other sentiment analysis models. The experimental outcomes suggest a better performance of the proposed model than the other methods, in terms of accuracy and F-measure, for sentiment analysis of student feedback.

### 2.2 Deep Learning Techniques for sentimental analysis

Deep learning has been widely used in the field of natural language processing for various tasks, including sentiment analysis and opinion mining. In recent years, many researchers have proposed different deep-learning architectures for these tasks, particularly for the analysis of student feedback.
One popular architecture uses convolutional neural networks (CNNs) for sentiment analysis. A study by Santos and Gatti [8] proposed a CNN-based approach for sentiment analysis of movie reviews and Twitter messages, showing that the model achieved state-of-the-art performance. Similarly, Kalch- brenner et al. [9] proposed a dynamic CNN architecture, which was also able to achieve high performance on sen- timent analysis tasks.
Another popular architecture uses recurrent neural networks (RNNs) and their variants, like LSTM and GRU, for sentiment analysis. A study by Tai et al. [10] proposed a hierarchical RNN model for sentiment analysis of social media texts. The authors showed that the proposed model outperformed existing methods on several benchmark datasets. Similarly, a study by Wang et al. [11] proposed a bidirectional LSTM model for sentiment analysis of student feedback, which achieved high performance.
The sentiment analysis hybrid deep learning models that integrate convolutional neural networks (CNN), long short-term memory (LSTM) networks,and support vector machines (SVM) are developed and tested in [12] on eight review datasets and textual tweets of various domains. A comparison is done between the hybrid models and three single models, SVM, LSTM, and CNN. Both reliability and computation time was con-

sidered in the evaluation of each technique.
Recently, Transformer-based architectures like BERT and GPT-2 have also been widely used for opinion mining and sentimental analysis tasks. A study by Devlin et al. [13] proposed BERT, a transformer-based model that achieved state-of-the-art performance on several natural language understanding tasks, including sentiment analysis. Similarly, a study by Radford et al. [14] proposed GPT-2, a transformer-based language model that could generate human-like text and showed its ability in opinion mining.

## 3 Methodology

### 3.1 About Dataset

For this research, we have taken the dataset from the website RateMyProfessors.com, an online platform that helps educational organizations take feedback from students regarding their experience with their course teachers. It allows students to express opinions regarding the course content or any suggestion they would like to give to the administration for improving the quality of education. On the other hand, it allows teachers to look at the statistics generated from student ratings. It enables them to learn about the point of view of the students, which helps them improve their teaching methodology.
The dataset contains 19000 comments along with the individual student rating and the average rating.

Table 1: Student ratings

| Student-Rating | Average rating | Comments |
|---|---|---|
| 5 | 4.7 | Excellent professor. Hilarious, fun, and good |
| 1.6 | 1.5 | Warning: By far the worst online teacher no communication skills, no desire to help |
| 3.5 | 3.5 | One word. DRY. |

The comments were categorized into three categories positive, neutral, and negative, based on student ratings ranging between 0 to 5. If the rating was higher than 3.5, it was rated as positive. If it was below 2.5, it was rated as negative and if it was between 2.5 and 3.5, it was rated neutral. In this manner, 11200 positive comments, 4400 negative comments, and 3400 neutral comments were gathered. The positive comments were down-sampled, and the rest of the categories were up-sampled. Models were trained and tested on this dataset and then were further tested on another dataset containing 100 comments collected within the classes at our institution.

### 3.2 Proposed Framework

Figure 1 shows the framework that we adopted for this research. The sentiment analysis process after data collection was carried out in several stages.

The first stage involved pre-processing the text data by removing numbers, and punctuation, conv-erting to lowercase, tokenizing, removing stop words, and lemmatization to make the data more consistent and noise free. In the next stage, we used Feature Extraction Techniques (FETs) such as Bag of words (BoW), term frequency with inverse document frequency (TF-IDF), global vector (GloVe), and word- to-vector (word2vec) to extract meaningful features from the student dataset. The data was then partitioned using the tensor flow library into training and testing sets. The experiment was further divided into three stages. Firstly, the machine learning algorithms were applied for training and testing the model using the BoW, TF-IDF, and word2vec features, and their performance was recorded. In the second stage, the model was trained with a hybrid of machine learning algorithms and ensemble techniques and got re-evaluated using the same three features. Finally, deep learning architecture was used to train and test the model using word2vec and GloVe embedding. Their performance was evaluated against various performance metrics.

### 3.3 Preprocessing

The student feedback is obtained in the English language as natural language. The machine learning model cannot process text input, so we must convert it into a format it can understand. Preparing the data is the first step in this process, as text data is challenging due to noise, such as incorrect punctuation, slang, emoticons, and spelling mistakes. Words like" BOO,"" prof," and "tooooo much" can confuse the model, and everyday words like "they," "you," and "he" are known as stop words, which do not convey any emotion. To improve the accuracy of the model, these stop words should be removed. The pre-processing steps applied to the dataset include:

- removing numbers and punctuation,

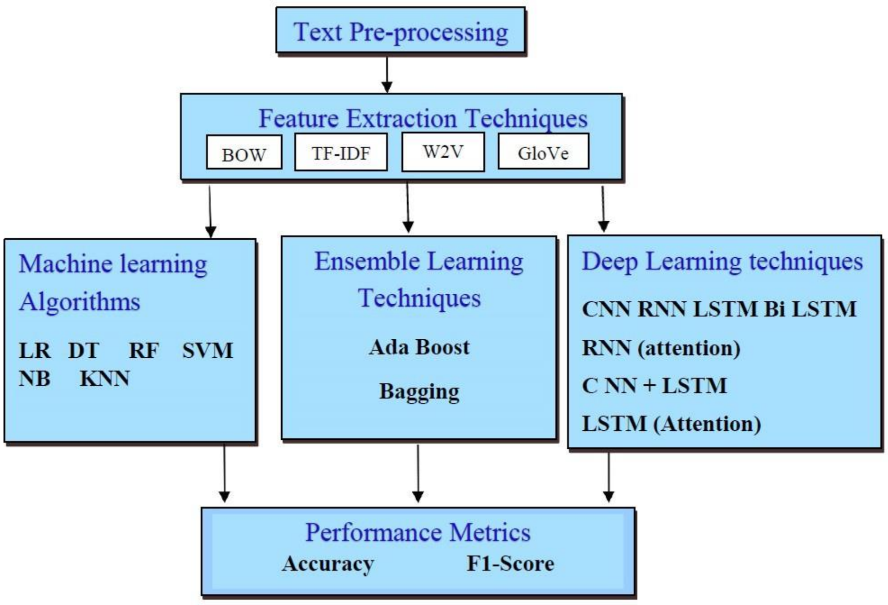

Figure 1: Proposed Framework

- replacing contraction
- converting characters to lowercase
- tokenization,
- removing stop words,
- removing no comments,

For example, the following Table 2 shows an example of the original comment and the corresponding pre-processed comment: -

Table 2: Original and pre-processed comment

| | |
|---|---|
| Original Comment | I overall enjoyed this class because the assignments were straightforward and interesting. I just did not enjoy the video project because I felt like no one in my group cared enough to help |
| Preprocessed Comment | overall enjoyed class assignments straightforward interesting not enjoy video project felt like no one group cared enough help |

## 3.4 Feature Extraction

In this research, we have used conventional text representation and pretrained word embedding. For machine learning algorithms, we have used a bag of words, term frequency with inverse document frequency, and word-to-vector technique. For deep learning architecture, we have used term frequency with inverse document frequency, word to vector technique, and lastly, global vector word embedding.

### 3.4.1 Bag of Words (BoW)

Bag-of-words (BoW) [15] is a method of feature extraction for natural language processing and information retrieval that represents text as a bag or set of words where each word is a feature and the frequency of the word in the text is the value of the feature. In the BOW model, each document is represented by a vector, where each entry in the vector corresponds to a word in a pre-defined vocabulary, and the value of the entry is the number of times that word appears in the document. In the case of unique words will be represented by a vector with a value of 1 in the corresponding entry in the vocabulary and 0 in the rest of the entries. For example, if the vocabulary is {'He', 'explains', 'the', 'concepts', 'very', 'well', 'but', 'not', 'algorithm'} and the document is "He explains the concepts very well but not explains the algorithm", the BoW for this document is {1, 2, 2, 1, 1, 1, 1, 1, 2, 2, 1}.
The main benefit of using the BoW model is that it's simple to implement and computationally efficient, which makes it suitable for large-scale text classification tasks. However, it does not consider the context or order of the words in the text, which can be important for understanding the meaning and sentiment of the text. On the other hand, the BoW model could be better at dealing with words that are rare or unseen in the training data, as they will be represented by a zero vector, which can cause issues in the model's ability to generalize to new data. To overcome this, techniques such as n-grams and TF-IDF can be used to consider the context and rarity of words.

### 3.4.2 Term frequency-Inverse document frequency (TF-IDF)

Term Frequency-Inverse Document Frequency (TF-IDF) [15] is a feature extraction method for natural language processing and information retrieval that assigns a weight to each word in a document based on its frequency and in the entire corpus of documents.
The term frequency (TF) measures how often a word appears in a document, assuming that more frequent words are more informative and relevant to the document's content. The inverse document frequency (IDF) measures how rare a word is across the entire corpus, assuming that rarer words are more informative and specific to a document's content.
The TF-IDF weight for a word in a document is calculated as the product of its term frequency and its inverse document frequency:

$$TF - IDF = TF(word, document) \times IDF(word) \quad (1)$$

$$TF(word, document) = \frac{NOWD}{TWD} \quad (2)$$

where
NOWD=No. of occurrences of word in the document
TWD = Total no. of words in the document

$$IDF(word) = log(\frac{TND}{ND}) \quad (3)$$

TND = Total number of documents
ND = Number of documents containing the word
The unique words (or unique terms) are the words that appear only once in a document, these are also known as hapax legomena. In the TF-IDF weighting, unique words will have a weight of log(N) where $N$ is the number of documents in the corpus, this is because the unique words are present in only one document so the IDF will be log(N/1) = log(N).
One of the main benefits of using TF-IDF is that it can help to reduce the dimensionality of the feature space by giving less importance to common and less informative words, such as stop words, and more importance to rare and informative words. This can improve the performance of machine learning models for text classification and other natural language processing tasks.

### 3.4.3 n-gram

In natural language processing, an n-gram [16] is a contiguous sequence of n words. n-grams are used as features in text-based feature extraction, where they capture the context and order of words in the text. When using N-grams as features, the goal is to capture the meaning of the text in a way that reflects the context in which the words appear. For example, the word "play" can have different meanings depending on the context, and by including the preceding and following words in the form of bigrams or trigrams, it can be possible to capture the meaning more accurately.
N-grams are also helpful in handling cases where the context of a word is essential, such as idioms, collocations, and phrasal verbs. For example, the bigrams "kick the" and "the bucket" would not have much meaning separately, but together they form the idiomatic expression "kick the bucket" which means dying.
It's important to note that while n-grams can be useful in capturing context, they also increase the dimensionality of the data and the computational cost of processing it, so a trade-off needs to be made between capturing more context and the complexity of the model. For our research, we have used unigram as it proved to be more resourceful than the other combinations.
Although a simple method for NLP tasks, the BoW scheme has two main shortcomings: First, the semantic link between the words in a document needs to be fully modeled in this method. Second, it results in a data representation which is sparse along with high dimensional feature space. To remedy the situation, machine-learning algorithms adopt a word embedding scheme [17], which is more expressive and compact.

### 3.4.4 Global vectors

Global Vectors (GloVe) is a word embedding method developed by researchers at Stanford University that represents words as high-dimensional vectors in a continuous vector space. These vectors capture the meaning, and context of words in a way that can be used in natural language processing tasks such as sentiment analysis, text classification, and machine translation. The GloVe model learns word embeddings by training on a large corpus of text data and using matrix factorization. The model takes a word-

word co-occurrence matrix and counts the number of times each word appears in the same context as every other word in the corpus as input. The model then factorizes this matrix into two lower-dimensional matrices, one representing the words and the other representing the context. The dot product of these two matrices results in the word vectors, which are used as the word embeddings.
The resulting word vectors have several desirable properties, capturing semantic and syntactic similarity between words. It allows the model to make inferences about the meaning of words based on their vector representation. For example, vectors for words like “king” and “queen” will be close to each other, while vectors for” king” and” car” will be farther apart.

#### 3.4.5 Word2Vec

Word2Vec is a word embedding technique that represents words as dense vectors in a continuous space, capturing the semantic meaning and context of words. It uses a neural network architecture to predict the context words given a target word, and the network weights are used as the word vectors [18].
Two variants of word2vec are Continuous Bag-of-Words (CBOW) and Skip-Gram. CBOW predicts target words from their context words, while Skip-Gram predicts context words from a target word. Both models are trained on a large corpus of text, and the learned word vectors are then used in various NLP tasks such as text classification, sentiment analysis, and machine translation [18].
Word2Vec is widely used because of its ability to capture the semantic meaning of words and the relationships between words in a sentence. It has also been shown to produce high-quality word embeddings that can be used as features in various NLP models, leading to improved performance [19].

## 3.5 Machine learning Algorithms

In the research study, the pre-processed data was trans- formed into different feature sets using unigram and weighting schemes (BOW, TF-IDF, word2vec). Then six conventional machine learning algorithms [20, 21] (Decision Forest, Naive Bayes, Support Vector Machines, Logistic Regression, K-Nearest Neighbors, and Random Forest) were applied to build learning models based on these feature sets. Following algorithms are used to train the dataset:

#### 3.5.1 Naive Bayes (NB)

It is a popular machine learning algorithm for classification problems. It is based on Bayes’ theorem and assumes that the features of a data point are independent of each other, hence the name” Naive”. It is simple to implement and efficient in terms of computation time, making it a good choice for large datasets. The algorithm calculates the probability of a data point belonging to each class, and the class with the highest probability is assigned as the output. Naive Bayes has been successfully used in text classification, sentiment analysis, and spam filtering [22].

#### 3.5.2 Decision Tree (DT)

It is a tree-based model for supervised learning problems that can be used for classification and regression. It uses a tree-like structure to represent decisions and their possible consequences. The algorithm starts by choosing the feature that best separates the data into classes. It then recursively applies the same process to the subsets of data until a stopping criterion is met. The result is a tree of decisions, where each node represents a feature, and each leaf represents a prediction. Decision trees are easy to interpret, handle missing values and non-linear relationships, and are suitable for multi-class problems [23].

#### 3.5.3 Random Forest (RF)

It is an ensemble learning method for classification and regression problems. It consists of multiple decision trees that operate as a collection to make predictions. Each tree is trained on a different random subset of the data and a random subset of the features. The final prediction is made by averaging the predictions of all trees. Random Forest is known for its good performance and ability to handle extensive and high-dimensional data. It is also less prone to overfitting compared to single decision trees and provides a measure of feature importance. The algorithm is widely used in various fields, such as medicine, finance, and ecology [24].

#### 3.5.4 Logistic Regression (LR)

It is a statistical method for solving binary classification problems. It uses a logistic function to model the relationship between the dependent variable and one or more independent variables. The logistic function maps the predicted values to a probability between 0 and 1, which can then be used to classify data points into one of two classes. Logistic Regression is easy to interpret, fast to train, and can handle non-linear relationships by transforming the independent variables. It is widely used in medical research, marketing, and finance to predict binary outcomes, such as customer churn or disease diagnosis [25].

#### 3.5.5 Support Vector Machines (SVM)

It is a popular algorithm for solving classification and regression problems. It works by finding the hyperplane that maximally separates the data into classes [26]. SVM is known for its ability to handle non-linear relationships and high-dimensional data using kernel functions [27]. It is also effective in handling small datasets, as it is less prone to overfitting than other algorithms [28]. SVM has been successfully applied in various fields, such as bioinformatics, computer vision, and natural language processing. The algorithm has the advantage of being able to handle both linear and non-linear problems, making it a versatile tool for data analysis [29].

#### 3.5.6 K-Nearest Neighbors (KNN)

It is a simple, instance-based, and lazy learning algorithm for classification and regression problems. It works by finding the K nearest data points to a given test point and making a prediction based on the majority class or average value of those K nearest points [30]. KNN is effective in handling small datasets and is useful for problems with high-dimensional data, as it does not require explicit computation of feature weights [31]. The algorithm is fast during testing, as it only requires finding the K nearest neighbors. KNN has been applied in various fields, such as computer vision, finance, and recommendation systems [32]. It is a good choice for problems where the relationships between features need to be understood or are non-linear [31].

### 3.6 Ensemble Learning Models

Ensemble learning is a method where multiple supervised learning models work together to make a prediction [33]. The idea is that combining these models will result in a more robust and accurate prediction. In the past, research has shown that ensemble learning can be effective in sentiment analysis. The current study uses two ensemble learning methods (AdaBoost and Bagging) with six supervised learning algorithms.

AdaBoost is an ensemble learning method that trains the base models sequentially and builds a new model at each round. The idea is that the algorithm will dedicate more rounds to instances that are harder to learn, and correct errors made by previous models [34].

Bagging (Bootstrapped Aggregating) is an ensemble machine learning technique used to reduce the variance in the prediction of models by combining the output of multiple models. It works by training multiple models on different randomly selected subsets (with replacement) of the training data [35]. The final prediction is typically made by taking the average or majority vote of the individual models' predictions. Bagging helps to reduce overfitting in single models and can lead to improved performance on unseen data [36].

### 3.7 Deep learning Models

Deep learning is a subfield of machine learning that focuses on using artificial neural networks with multiple layers (hence the term" deep") to learn complex representations of data [37]. On the other hand, machine learning is a broader field that encompasses various algorithms and techniques for training models to make predictions or decisions without being explicitly programmed. It includes shallow algorithms like linear regression. logistic regression, and deep learning algorithms. The key difference between deep learning and other forms of machine learning lies in the depth of the model and the way it learns from data. Unlike traditional machine learning algorithms that rely on manually crafted features, deep learning algorithms learn a hierarchical representation of the data through multiple non-linear transfomations, allowing them to learn complex features and make highly accurate predictions automatically. Deep learning has proven highly effective for tasks such as image recognition, natural language processing, and speech recognition, among others [37,38].

### 3.7.1 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are a type of deep learning algorithm particularly well-suited for image and video recognition tasks [39]. The basic idea behind CNNs is to learn a set of convolutional filters that can extract useful features from the input data, such as edges, textures, and shapes.

CNN consists of several layers, including convolutional layers, pooling layers, and fully connected layers [39,40]. The convolutional layers are responsible for learning the convolutional filters and applying them to the input data to extract features. The pooling layers are used to reduce the dimensionality of the feature maps and to increase the robustness of the features to small translations and deformations. The fully connected layers are used to classify the input data based on the learned features [39]. One of the unique characterristics of CNNs is that they use a technique called weight sharing, which allows them to learn translation-invariant features, meaning that they are robust to small translations and deformations of the input data. It is particularly useful for image and video recognition tasks where objects of interest may appear in different positions. Another unique characteristic of CNNs is that they use a technique called pooling, which reduces feature maps' dimensionality and increases the features' robustness to small translations and deformations. This allows the model to focus on the most important features and ignore irrelevant or redundant information [39,40].

### 3.7.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) is a type of deep learning algorithm that are particularly well-suited for sequence-to-sequence tasks, such as natural language processing, speech recognition, and time series forecasting [41]. The basic idea behind RNNs is to introduce feedback connections in the neural network architecture, allowing the network to maintain information from previous time steps and use it to inform its predictions at the current time step.

An RNN consists of a sequence of recurrent layers, each of which takes as input the current input and the hidden state from the previous time step [42]. The hidden state is a vector that encodes the information that the network has learned so far and it is updated at each time step. The output of the RNN is generated by the final recurrent layer, which maps the hidden state to the final output.

One of the unique characteristics of RNNs is that they can maintain information from previous time steps and use it to inform their predictions at the current time step. It is particularly useful for sequence-to-sequence tasks, where the output at the current time step depends on the input and the output at previous time steps [41]. Another unique characteristic of RNNs is that they can handle variable-length sequences, unlike traditional feedforward neural networks designed to handle fixed-length inputs [42]. It allows RNNs to process sequences of different lengths without padding or truncation.

### 3.7.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) that are particularly well-suited for sequence-to-sequence tasks, such as natural language processing and speech recognition. The basic idea behind LSTMs is to introduce a set of memory cells and gates into the RNN architecture, allowing the network to selectively store and retrieve information from previous time steps and use it to inform its predictions at the current time step [43].

An LSTM network consists of a sequence of LSTM layers, containing a set of memory cells and gates. The gates control the flow of information into and out of the memory cells, allowing the network to store and retrieve relevant information from previous time steps selectively. The output of the LSTM network is generated by the final LSTM layer, which maps the hidden state to the final output [43].

One of the unique characteristics of LSTMs is that they can selectively store and retrieve information from previous time steps, allowing them to overcome the problem of vanishing or exploding gradients that can occur in traditional RNNs. It allows them to handle long-term dependencies better and make more accurate predictions [44]. Another unique characteristic of LSTMs is that they can handle variable-length sequences, unlike traditional feedforward neural networks, which are designed to handle fixed-length inputs. It allows LSTMs to process sequences of different lengths without the need for padding or truncation [43].

### 3.7.4 LSTM with an attention mechanism

LSTM (Long Short-Term Memory) is a type of Recurrent Neural Network (RNN) architecture designed to handle the problem of vanishing gradients in RNNs [45]. An LSTM network comprises of memory cells, gates (input, forget, and output gates), and fully connected layers [46]. The attention mechanism is a method for selectively focusing on specific parts of the input sequence when making predictions. It enables the model to weigh the importance of each part of the sequence differently and to focus on the most relevant information [47].

In the context of Natural Language Processing (NLP), the attention mechanism can be used to weigh the importance of different words in a sentence and to focus on the most relevant words when making predictions [47]. When combined with an LSTM network, the attention mechanism allows the model to focus dynamically on the most relevant parts of the input sequence for each prediction [48]. It allows the LSTM network to make predictions based on a weighted combination of information from different parts of the input sequence [48]. It has been shown to significantly improve the performance of NLP models on various tasks such as machine translation and text classification.

Combining LSTM with an attention mechanism provides a powerful tool for learning long-term dependencies in data sequences [47].

### 3.7.5 CNN combined with LSTM

Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are two popular deep learning architectures used for text classification tasks. A combination of these two architectures, known as CNN-LSTM, has been shown to improve the accuracy of text classification models. The CNN-LSTM model applies convolutional filters to extract high-level features from the input text data and then passes the output through a sequence of LSTM cells to capture the temporal dependencies in the data. It allows the model to learn local and global features from the input text, leading to better classification performance.

Several studies have demonstrated the effectiveness of the CNN-LSTM model on various text classification tasks, including sentiment analysis, topic classification, and spam detection. For example, [49] used a CNN-LSTM model to classify sentiment in movie reviews, achieving state-of-the-art performance on the dataset.

### 3.7.6 RNN with an attention mechanism

Recurrent Neural Networks (RNNs) are a type of neural network commonly used for processing sequential data. One of the significant challenges in processing sequential data is the ability to capture dependencies between different parts of the sequence. One solution to this problem is the use of attention mechanisms [50].

Attention mechanisms are a type of mechanism that allows the model to focus on specific parts of the input sequence when making predictions. It is done by assigning a weight to each input element based on its relevance to the prediction. The weights are learned during training and are used to compute a weighted sum of the input elements, which is then used as the input to the next layer of the model [51].

In the case of RNNs with attention, the attention mechanism is used to compute a weighted sum of the hidden states of the RNN. It allows the model to selectively attend to specific parts of the input sequence, rather than simply relying on the final hidden state of the RNN [52].

The use of attention mechanisms in RNNs has been shown to improve performance on a wide range of tasks, including machine translation, speech recognition, and image captioning [53]. The ability to sele-ctively attend to specific parts of the input sequence allows the model to capture complex dependencies between different parts of the sequence, leading to improved performance.

## 3.8 Evaluation Metric

In evaluating machine learning and deep learning algorithms, are classification accuracy (ACC) and F1-score are two common metrics used. These measures have been widely employed to assess the performance of predictive models.

### 3.8.1 Accuracy

Accuracy is a widely used evaluation measure in machine learning and deep learning. It is used to quantify the performance of a model in terms of the number of correct predictions it makes. It is defined as the ratio of number of correct predictions made by a model to the total number of predictions [54]. It can be calculated as follows:

$$Accuracy = \frac{Numver\ of\ correct\ predictions}{Total\ number\ of\ predictions} \quad (4)$$

Accuracy is a simple and intuitive measure and is particularly applicable when the classes in the target variable is balanced. However, it can be misleading when the classes are imbalanced, as model predicts, the majority class can achieve high accuracy. In such cases, other evaluation measures, such as precision, recall, and F1-score, should be used to understand the model's performance better.

### 3.8.2 F1-Score

The F1-score [55] is a measure of a model's accuracy that balances precision and recall. It is a harmonic mean of precision and recall and can be calculated as follows:

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (5)$$

Where Precision is the number of true positive predictions made by the model divided by the sum of true positive predictions and false positive predictions, and Recall is the number of true positive predictions made by the model divided by the sum of true positive predictions and false negative predictions.
The F1-score provides a balance between precision and recall. It is instrumental when the classes in the target variable are imbalanced or when the cost of false negatives and false positives is unequal. In such cases, simply optimizing for accuracy can result in a model that prioritizes one type of error over the other, which may not be desirable.

# 4 Experiment And Result

After pre-processing the data and feature extraction, the experiment was conducted in three batches. Firstly, data were trained on six different supervised machine-learning algorithms: three feature extraction techniques, namely BoW, TF-IDF, and word2vec, in combination with the unigram model. Labeling techniques like encoding were used to encode the sentiments of each comment. In the second batch, ensemble techniques (AdaBoost and Bagging) were used with a machine-learning algorithm to boost their performance. Five deep-learning architectures were deployed in the last batch with two word embedding techniques.

This section presents the results of classification accuracy and F1-Score, as produced by traditional supervised learning techniques and ensemble learning algorithms. Tables 3 and 4 represent the accuracy and F1-Score values obtained from machine learning algorithms for the given dataset.

Table 3: Accuracy of machine learning model

| Algorithms | BOW | TF-IDF(unigram) | W2V |
|---|---|---|---|
| DT | 75.65 | 74.70 | 76.26 |
| RF | 82.10 | 83.31 | 85.53 |
| NB | 71.00 | 76.55 | |
| KNN | 65.26 | 69.90 | 80.24 |
| LR | 74.37 | 76.00 | 66.66 |
| DT (Ada boost) | 78.00 | 77.72 | 87.33 |
| RF (Ada boost) | 89.93 | 84.23 | 90.12 |
| SVM (Ada boost) | 73.45 | 72.25 | 77.89 |
| LR (Ada boost) | 79.56 | 73.33 | 65.99 |
| DT (bagging) | 82.22 | 80.89 | 83.35 |
| RF (bagging) | 82.34 | 82.17 | 83.67 |
| SVM (bagging) | 81.11 | 82.12 | 82.21 |
| KNN (bagging) | 67.79 | 80.01 | 80.90 |
| LR (bagging) | 76.69 | 78.99 | 82.11 |

Table 4: F1-Score of machine learning model

| Algorithms | BOW | TF-IDF(unigram) | W2V |
|---|---|---|---|
| DT | 0.76 | 0.72 | 0.76 |
| RF | 0.82 | 0.83 | 0.85 |
| NB | 0.71 | 0.76 | |
| KNN | 0.63 | 0.70 | 0.80 |
| LR | 0.80 | 0.75 | 0.66 |
| DT (Ada boost) | 0.78 | 0.77 | 0.86 |
| RF (Ada boost) | 0.90 | 0.83 | 0.90 |
| SVM (Ada boost) | 0.75 | 0.72 | 0.77 |
| LR (Ada boost) | 0.79 | 0.73 | 0.66 |
| DT (bagging) | 0.78 | 0.83 | 0.83 |
| RF (bagging) | 0.83 | 0.83 | 0.83 |
| SVM (bagging) | 0.81 | 0.82 | 0.82 |
| KNN (bagging) | 0.67 | 0.80 | 0.80 |
| LR (bagging) | 0.75 | 0.79 | 0.82 |

## 4.1 Machine Learning

This study analyzed the classification accuracy of various supervised learning algorithms and ensemble methods when applied to three different configurations obtained from three text representation schemes of the dataset. The algorithms studied were Random Forest (RF), Naive Bayes (NB), Support Vector Machines (SVMs), Logistic Regression (LR), K-Nearest Neighbours (KNN), and Decision Tree (DT). Results showed that RF achieved the highest accuracy, followed by DT and SVMs. The word2vec with the vector size of 100 and 200 had the best accuracy, followed by BoWs in second place and unigram features with TF-IDF weighting coming in third. The study found that word embedding models outperformed classic feature extraction techniques. However, RF applied with BoW almost gave the same accuracy as with word2vec.

The study also looked at the impact of ensemble methods, including AdaBoost, and Bagging, on the predictive performance of the supervised learning methods. First, a base machine learning algorithm such as a decision tree, logistic regression, or support vector machine is chosen. Next, multiple bootstrap samples of the training set are created by randomly selecting training examples with replacements. Each of these bootstrap samples is given equal weight, and a base model is trained on the weighted training set. The error rate of the base model on the training set is then computed. In AdaBoost, the weights of the misclassified examples are increased, while the weights of the correctly classified examples are decreased. It puts more emphasis on the examples that are difficult to classify. Training a base model, computing the error rate, and updating the weights of the training examples are repeated for a fixed number of iterations. Finally, the outputs of the base models are combined by weighing them according to their error rates, and the resulting ensemble model is evaluated on a test set. The results showed using ensemble methods improved the accuracy of the supervised learning algorithms. The highest accuracy among the configurations studied was obtained by the AdaBoost ensemble of RF, with a classification accuracy of 90.12%. In summary, RF performs better than other machine learning algorithms when combined with AdaBoost for text classification because it reduces the variance of the model, while AdaBoost reduces the bias. This combination allows the resulting model to make accurate predictions on high-dimensional and noisy datasets.

Overall, the results of this study provide valuable insights into the accuracy of various supervised learning algorithms and ensemble methods when applied to text corpus configurations. The findings can be helpful for practitioners in choosing the most appropriate machine learning algorithm for their text classification task, as well as for researchers in developing new approaches for text classification.

## 4.2 Deep Learning

This research used deep learning-based sentiment analysis on a text-based dataset. To represent the textual comments, two word embedding schemes were used, namely, word2vec and GloVe, as mentioned in the previous section. Six deep learning architectures were considered, including CNN, RNN, bidirectional RNN-AM, long short-term memory (LSTM), CNN combined with LSTM, and bidirectional LSTM for processing the text data.

These deep learning architectures were implemented and trained using different libraries such as TensorFlow and Keras. The optimal performance for each model was obtained through hyperparameter optimization based on Bayesian optimization with a Gaussian process. 80% of the data was used as the training set and validation set, while the rest was used as the testing set. In the case of word2vec, both continuous skip-gram and CBOW schemes were considered, with varying vector sizes and dimensions of projection layers. The general structure of deep learning-based sentiment analysis is summarized in Figure 1.

Table 5: Accuracy pf deep learning techniques

| Algorithms | Vector Size | W2Vec | Glove |
|---|---|---|---|
| CNN | 200 | 80.13 | 82.13 |
| RNN | 200 | 84.02 | 84.22 |
| LSTM | 200 | 73.63 | 75.63 |
| Bidirectional LSTM | 200 | 80.22 | 80.22 |
| Bidirectional LSTM with Attention mechanism | 200 | 87.52 | 87.52 |
| CNN with LSTM | 200 | 89.45 | 90.22 |
| CNN | 300 | 79.87 | 81.99 |
| RNN | 300 | 85.06 | 85.06 |
| MLP | 300 | 78.09 | 78.09 |
| Bidirectional LSTM | 300 | 80.03 | 81.99 |
| Bidirectional RNN with Attention mechanism | 300 | - | 90.76 |
| Bidirectional LSTM with Attention mechanism | 300 | 83.31 | 87.35 |
| CNN with LSTM | 300 | 88.07 | 91.29 |

Table 5 presents the classification accuracy obtained by the seven deep learning architectures.

The results from the empirical analysis showed that the GloVe word embedding scheme outperformed the other word embedding schemes for the text corpus. The word2vec skip-gram model obtained the lowest predictive performance. The results indicated that the best predictive performance was achieved for the vector size of 300 and the dimension projection layer of 300. The highest predictive performance among the deep learning architectures was achieved by CNN combined with cascaded layers of LSTM followed by the bidirectional RNN-AM. These findings can contribute to the advancement of deep learning-based sentiment analysis in text analytics.

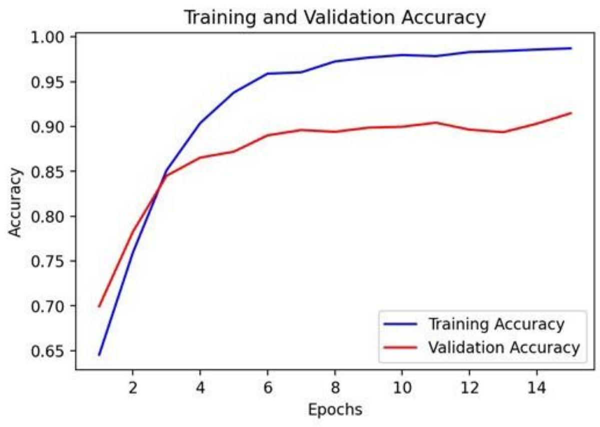

Figure 2: Training and Validation Accuracy

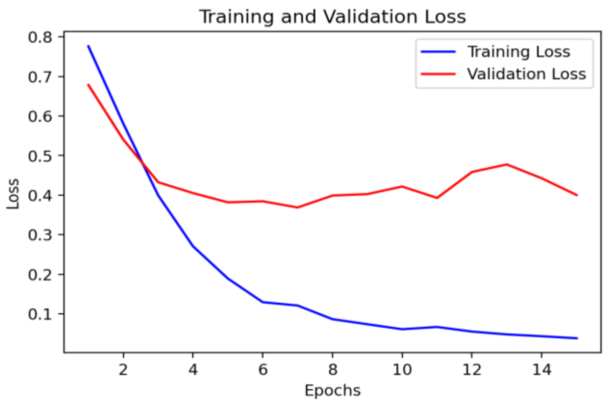

Figure 3: Training and Validation loss

The number of epochs is a hyperparameter that detrmines how many times the model will be trained on the dataset. Generally, the more epochs a model is trained for, the better it will perform on the training data. However, training for too many epochs may cause overfitting, where the model performs well on the training data but poorly on new, unseen data. Figure 2 represents training and validation accuracy; we see a steep rise in training accuracy, whereas testing data accuracy improved slowly but surely. As seen in Figure 2, training accuracy reaches 1, but the validation accuracy reaches up to 92%. The reason for the halted progress can be seen in Figure 3 as training loss reduces, validation loss becomes stagnant, indicating that the model is overfitted to the training data.

After experimenting with three batches, the model that provided the best accuracy was chosen to predict sentiments from one hundred comments collected within university premises. In this case, CNN with LSTM cascaded architecture was chosen. It assigned probabilities to all three classes, and the class with the highest probability was selected as the predicted sentiment. When a sentence was passed to the model, it went through pre-processing steps first and then outputted the predicted sentiment (positive, negative, or neutral) based on the highest probability in the final output vector.

## 4.3 Discussion

The study evaluated various machine learning techniques including standard classifiers, ensemble methods, and deep learning models [55]. Results showed that ensemble methods generally have better performance over traditional classifiers, while deep learning models outperformed ensemble techniques. The highest classification accuracy of 91.29% was achieved by using a cascaded Neural Network of CNN combined with LSTM, followed closely by Bi-directional Recurrent Neural Network Mechanism (RNN-AM) with a Global Vectors (GloVe) word embedding representation [55].

The experimental outcomes suggest that deep learning frameworks have the potential to produce noteworthy results in education for tasks related to machine learning and data mining [55]. It is worth noting that conventional machine learning techniques produced similar results and consumed less computational resources and time than deep learning architecture. Using other embedding techniques may produce better results, and the possibility of outperforming other deep learning techniques presides [55].

The dataset used in the empirical analysis was gathered from Ratemyprofessors.com [55], where most contributors are from instructors and schools in the USA. This specific dataset used in the study may impact the results of machine learning and data science. Most of the data set consisted of positive comments due to which we had to down sample the comments as it affected our accuracy. This is considered a constraint of the research. However, the method used in the study follows a machine learning-based approach for text sentiment classification and can be applied for sentiment analysis in other languages with proper pre-processing [55].

## 5 Conclusion

This paper presents a text-mining method for analyzing instructor evaluation reviews. A dataset of 20,000 reviews was collected for the study. Out of which, 1000 were discarded for not providing helpful information. The analysis consisted of evaluating various machine learning algorithms, including conventional supervised methods (such as Decision tree, Naive Bayes, Support Vector Machines, Logistic Regression, k-Nearest Neighbors, and Random Forest), ensemble learning techniques (AdaBoost, and Bagging), and deep learning models (Convolutional Neural Network, Recurrent Neural Network, Bidirectional RNN with Attention Mechanism, Bidirectional LSTM and CNN combined with LSTM). The study utilized two conventional text representation techniques (BOW and TF-IDF) with conventional machine learning algorithms and two word embedding approaches (word2vec and GloVe) with deep learning models. The results showed that deep learning- based methods outperformed ensemble learning and supervised learning methods for sentiment classification. Among the configurations compared, the highest accuracy of 91.29% was obtained using a cascaded CNN combined with an LSTM in combination with a GloVe based word embedding representation.

## Declarations

” **Funding:** No funding
**Conflicts of interest/Competing interests:** None **Availability of data and material:** All data generated or analyzed during this study are included in this published article as different tables.
**Code availability:** All the code is developed in Python and is available with the authors
**Consent to participate:** Not applicable
**Ethics approval:** Not applicable
**Consent for publication:** Not applicable.

## References

[1] E. Cambria, D. Das, S. Bandyopadhyay, A. Feraco, *et al.*, “A practical guide to sentiment analysis,” 2017.

[2] J. Zhao, K. Liu, and L. Xu, “Sentiment analysis: mining opinions, sentiments, and emotions,” 2016.

[3] A. Onan, “Mining opinions from instructor evaluation reviews: a deep learning approach,” *Computer Applications in Engineering Education*, vol. 28, no. 1, pp. 117–138, 2020.

[4] L. Tamrakar, D. Shrivastava, D. S. Ghosh, *et al.*, “Student sentiment analysis using classification with feature extraction techniques,” *arXiv preprint arXiv:2102.05439*, 2021.

[5] K. Z. Aung and N. N. Myo, “Sentiment analysis of students’ comment using lexicon based approach,” in *2017 IEEE/ACIS 16th international conference on computer and information science (ICIS)*, pp. 149–154, IEEE, 2017.

[6] N. Altrabsheh, M. Cocea, and S. Fallahkhair, “Predicting learning-related emotions from students’ textual classroom feedback via twitter.,” *International Educational Data Mining Society*, 2015.

[7] Z. Nasim, Q. Rajput, and S. Haider, “Sentiment analysis of student feedback using machine learning and lexicon based approaches,” in *2017 international conference on research and innovation in information systems (ICRIIS)*, pp. 1–6, IEEE, 2017.

[8] C. Dos Santos and M. Gatti, “Deep convolutional neural networks for sentiment analysis of short texts,” in *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers*, pp. 69–78, 2014.

[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences,” *arXiv preprint arXiv:1404.2188*, 2014.

[10] K. S. Tai, R. Socher, and C. D. Manning, “Improved semantic representations from tree-structured long short-term memory networks,” *arXiv preprint arXiv:1503.00075*, 2015.

[11] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, “Attention-based bidirectional long short-term memory networks for relation classification,” in *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pp. 207–212, 2016.

[12] C. N. Dang, M. N. Moreno-Garc´ıa, and F. De la Prieta, “Hybrid deep learning models for sentiment analysis,” *Complexity*, vol. 2021, pp. 1–16, 2021.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[14] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[15] G. Hackeling, *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd, 2017.

[16] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, “The influence of preprocessing on text classification using a bag-of-words representation,” *PloS one*, vol. 15, no. 5, p. e0232525, 2020.

[17] N. Alami, M. Meknassi, and N. En-nahnahi, “Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning,” *Expert systems with applications*, vol. 123, pp. 195–211, 2019.

[18] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[19]Goldberg, Y., & Levy, O. (2014). Word2vec Explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722.

[20] L. Breiman, “Bagging predictors,” Machine learn- ing, vol. 24, pp. 123–140, 1996.

[21] L. Breiman, “Random forests,” *Machine learning*, vol. 45, pp. 5–32, 2001.

[22] Naive Bayes. (n.d.). In Wikipedia. Retrieved April 30, 2023, from https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[23] Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. Wadsworth International Group

[24] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[25] Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression. John Wiley & Sons. Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152). ACM.

[26] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152). ACM

[27] Schölkopf, B., & Smola, A. J. (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning). The MIT Press.

[28] Bishop, C. M. (2006). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc.

[29] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

[30] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-27.

[31]classification. IEEE Transactions on Information Theory, 13(1), 21-27.

[32] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3), 175-185.

[33] Zhang, Z. (2010). A review on k-nearest neighbor search algorithms. ACM Computing Surveys, 43(6), 1-28.

[34] J. R. Quinlan, "Bagging, boosting, and C4.5," in Proceedings of the Thirteenth National Conference on Artificial Intelligence, 1996, pp. 725-730.

[35] L. Breiman, "Arcing classifiers (with discussion)," in The Annals of Statistics, vol. 26, no. 3, pp. 801-849, 1998.

[36] L. Breiman, "Bagging predictors," in Machine Learning, vol. 24, no. 2, pp. 123-140, 1996.

[37] T. K. Ho, "The random subspace method for constructing decision forests," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 8, pp. 832-844, Aug. 1998.

[38] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[39] Alpaydin, E. (2020). Introduction to machine learning. MIT press.

[40] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[41] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[42] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555, 2014.

[43] A. Karpathy, "The unreasonable effectiveness of recurrent neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2442-2451.

[44] H. Sak et al., "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in Proc. Interspeech, 2014, pp. 338–342.

[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[46] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997.

[47] C. Olah, "Understanding LSTM Networks," Colah's Blog, 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: May 2, 2023].

[48] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," arXiv preprint arXiv:1409.0473, 2014.

[49] A. Vaswani et al., "Attention Is All You Need," in Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, Dec. 2017, pp. 6000-6010.

[50] Zhang, X., Zhao, J., & LeCun, Y. (2016). Character-level convolutional networks for text classification. In Advances in neural information processing systems (pp. 649-657). Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[51] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[52] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning (pp. 2048-2057).

[53] J. R. Quinlan, “Induction of decision trees,” Machine Learning, vol. 1, no. 1, pp. 81–106, 1986.

[54] Van Rijsbergen, C. J. (1979). Information Retrieval. Butterworths N. N. V. Kumar, R. Kumar, and K. Rani, "A Comparative Study of Machine Learning Techniques for Sentiment Analysis in Education Domain," in Proceedings of the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2021, pp. 498-502, doi: 10.1109/CONFLUENCE52294.2021.9483404.