# DEVELOPMENT OF DEEP LEARNING-BASED SENTIMENT ANALYSIS APPROACHES WITH NEURAL NETWORK-BASED LANGUAGE MODELS

**2023**
**MASTER THESIS**
**COMPUTER ENGINEERING**

**Khadija MOHAMAD**

**Thesis Advisor**
**Assist. Prof. Dr. Kürşat Mustafa KARAOĞLAN**

# DEVELOPMENT OF DEEP LEARNING-BASED SENTIMENT ANALYSIS APPROACHES WITH NEURAL NETWORK-BASED LANGUAGE MODELS

**Khadija MOHAMAD**

**Thesis Advisor**
**Assist. Prof. Dr. Kürşat Mustafa KARAOĞLAN**

**T.C.**
**Karabuk University**
**Institute of Graduate Programs**
**Department of Computer Engineering**
**Prepared as**
**Master Thesis**

**KARABUK**
**July 2023**

I certify that in my opinion the thesis submitted by Khadija MOHAMAD titled "DEVELOPMENT OF DEEP LEARNING-BASED SENTIMENT ANALYSIS APPROACHES WITH NEURAL NETWORK-BASED LANGUAGE MODELS" is fully adequate in scope and in quality as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Kürşat Mustafa KARAOĞLAN ..........................
Thesis Advisor, Department of Computer Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Computer Engineering as a Master of Science thesis. July 25, 2023

| <u>Examining Committee Members (Institutions)</u> | | <u>Signature</u> |
|---|---|---|
| Chairman | : Prof. Dr. Oğuz FINDIK (KBU) | .......................... |
| Member | : Assoc. Prof. Dr. Rafet DURGUT (BANU) | ONLİNE |
| Member | : Assist. Prof.Dr. Kürşat Mustafa KARAOĞLAN (KBU) | .......................... |

The degree of Master of Science by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabuk University.

Prof. Dr. Müslüm KUZU ..........................
Director of the Institute of Graduate Programs

*“I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well.”*

Khadija MOHAMAD

## ABSTRACT

**M. Sc. Thesis**

**DEVELOPMENT OF DEEP LEARNING-BASED SENTIMENT ANALYSIS APPROACHES WITH NEURAL NETWORK-BASED LANGUAGE MODELS**



**Khadija MOHAMAD**

**Karabük University**
**Institute of Graduate Programs**
**The Department of Computer Engineering**

**Thesis Advisor:**
**Assist. Prof. Dr. Kürşat Mustafa KARAOĞLAN**


**July 2023, 56 pages**

Sentiment Analysis (SA) is a Natural Language Processing task that automatically identifies and categorises sentiments or opinions expressed in textual sources. Artificial Intelligence techniques applied in SA tasks are essential in improving business decision-making by providing more precise and reliable insights into customer preferences and sentiment trends. In this context, large-scale product reviews are a valuable source of information with significant potential for businesses aiming to extract the distinctive features of their products, understand customer sentiments and thus improve their services. However, for businesses serving large user populations, performing accurate and precise analysis of product feedback is a complex process that requires automated approaches. Furthermore, difficulties can arise in verifying the consistency of user-generated satisfaction ratings with the relevant review. Another challenge in SA tasks is the inherent complexity of sentiment

expressions, where words or phrases can express different sentiments in various contexts. Given the complexity of sentimental expressions, interpreting words correctly requires understanding the context in which they are used. Therefore, contextualizing words or sentences using automated approaches and selecting appropriate classifiers are critical factors to perform better in solving SA problems.

To address the aforementioned problems, this thesis proposes Deep Learning-based SA approaches that use contextualized and context-free (static) language models as their inputs, using product reviews and user satisfaction ratings. In this thesis, state-of-the-art pre-trained language models in the form of Bidirectional Encoder Representations from Transformers (BERT) and Embeddings from Language Model (ELMo) are used to generate the contextual representation vectors of words or sentences to generate richer word or sentence representations to capture the sentiment expressed in textual sources more accurately. In addition to Contextualized Language Models (CLMs), Static Neural Network-based Language Models (SLMs) such as Word2Vec, Global Vectors for Word Representation (GloVe) and FastText are also used in this study. Classification models based on Deep Feed Forward Neural Networks, Long-Short Memory, Bidirectional Long-Short Memory, and Convolutional Neural Networks have been developed to classify word or text representations. To analyze the effectiveness of the proposed approaches and the contribution of the applied language models to the classification performance, experimental studies were carried out on Amazon review data, considered a benchmark dataset by most researchers in the literature. When the results of the experimental studies are analyzed, high and competitive performance results are obtained with the proposed approaches. In particular, the approach using the CNN-based BERT language model was found to have the highest performance, with 97% training and 95% testing accuracy.

In summary, it has been observed that Deep Learning-based SA approaches using CLMs and SLMs effectively capture sentimental expressions in textual sources. Moreover, the findings reveal the potential and practical value of the proposed approaches in developing SA techniques for businesses to decide on customer preferences and sentiments.

**Key Words :** Natural language processing, Deep Learning, Sentiment analysis, Contextual language models, Static language models, ELMo, BERT, Word2Vec, GloVe, FastText, DFFNN, LSTM, BiLSTM, CNN.

**Science Code :** 92416

## ÖZET

**Yüksek Lisans Tezi**

**SiNİR AĞI TABANLI DİL MODELLERİYLE DERİN ÖĞRENME TABANLI DUYUGU ANALİZİ YAKLAŞIMLARININ GELİŞTİRİLMESİ**

**Khadija MOHAMAD**

**Karabük Üniversitesi**
**Lisansüstü Eğitim Enstitüsü**
**Bilgisayar Mühendisliği Anabilim Dalı**

**Tez Danışmanı:**
**Dr. Öğr. Üyesi Kürşat Mustafa KARAOĞLAN**
**Temmuz 2023, 56 sayfa**

Duygu Analizi (DA), metinsel kaynaklarda ifade edilen duyguların veya fikirlerin otomatik olarak tanımlanmasını ve kategorize edilmesini içeren Doğal Dil İşleme görevidir. DA görevlerinde uygulanan Yapay Zeka teknikleri, müşteri tercihleri ve duygu eğilimleri hakkında daha kesin ve güvenilir içgörüler sağlayarak işletmelerin karar alma süreçlerini geliştirmede önemli bir rol oynamaktadır. Bu bağlamda büyük ölçekli ürün incelemeleri, ürünlerinin ayırt edici özelliklerini çıkarmayı, müşteri duygularını anlamayı ve bu sayede hizmetlerini iyileştirmeyi amaçlayan işletmeler için önemli bir potansiyele sahip değerli bilgi kaynağıdır. Ancak büyük kullanıcı popülasyonuna hizmet veren işletmeler için, ürünlerinin geri bildirimleri üzerinde doğru ve hassas analizler gerçekleştirmek otomatikleştirilmiş yaklaşımlar gerektiren karmaşık bir süreçtir. Bununla birlikte kullanıcıların oluşturduğu memnuniyet derecelendirmelerinin ilgili incelemeyle olan tutarlılığının doğrulanmasında da zorluklar ortaya çıkabilmektedir. DA görevlerinde diğer bir zorluk ise kelime veya

ifadelerin çeşitli bağlamlarda farklı duyguları ifade edebildiği duygu ifadelerinin doğasında var olan karmaşıklıktır. Duygu ifadelerindeki karmaşıklık göz önüne alındığında, kelimeleri doğru bir şekilde yorumlamak, ifadelerin kullanıldığı bağlamı anlamayı gerektirir. Bu sebeple DA problemlerinin çözümünü daha performanslı bir şekilde gerçekleştirmek için; otomatikleştirilmiş yaklaşımlar kullanarak, kelimeleri veya cümleleri bağlamsallaştırmak ve uygun sınıflandırıcıları seçmek kritik faktörlerdir.

Yukarıda bahsedilen problemleri ele almak için bu tez çalışmasında, ürün incelemeleri ve kullanıcı memnuniyet dereceleri kullanılarak, girdilerinde bağlamsallaştırılmış ve bağlamdan bağımsız (statik) dil modellerini kullanan Derin Öğrenme tabanlı DA yaklaşımları önerilmiştir. Söz konusu tez çalışmasında, daha zengin kelime veya cümle temsilleri üreterek metinsel kaynaklarda ifade edilen duyguların daha hassas bir şekilde ele alabilmek için, kelime veya cümlelerin bağlamsal temsil vektörlerinin üretilmesinde Dönüştürücülerden Çift Yönlü Kodlayıcı Temsilleri (BERT) ve Dil Modelinden Temsiller (ELMo) şeklinde son teknoloji önceden eğitilmiş dil modelleri kullanılmıştır. Bağlamsallaştırılmış Dil Modellerine (BDM) ek olarak bu çalışmada Word2Vec, Kelime Temsili için Küresel Vektörler (GloVe) ve Fasttext gibi Statik Sinir Ağı tabanlı Dil Modellerine (SDM) de yer verilmiştir. Kelime veya metin temsillerinin sınıflandırılmasında ise, Derin İleri Beslemeli Sinir Ağı, Uzun Kısa Süreli Bellek, Çift Yönlü Uzun Kısa Süreli Bellek ve Evrişimli Sinir Ağı tabanlı sınıflandırıcı modeller geliştirilmiştir. Önerilen yaklaşımların etkinliklerini ve uygulanan dil modellerinin sınıflandırma performanslarına olan katkılarını analiz etmek amacıyla, literatürde çoğu araştırmacı tarafından kıyaslama veri seti olarak kabul edilmiş Amazon inceleme verileri üzerinde deneysel çalışmalar gerçekleştirilmiştir. Deneysel çalışmaların sonuçları analiz edildiğinde, önerilen yaklaşımlarla yüksek ve rekabetçi performans sonuçları elde edilmiştir. Özellikle CNN tabanlı BERT dil modelini kullanan yaklaşımın %97 eğitim ve %95 test doğruluğu ile en yüksek performansa sahip olduğu belirlenmiştir.

Özetle BDM ve SDM kullanan Derin Öğrenme tabanlı Duygu Analizi yaklaşımlarının, metinsel kaynaklardaki duygu ifadelerinin ele alınmasında etkili performanslar sergilediği gözlemlenmiştir. Ayrıca bulgular DA tekniklerinin geliştirilmesinde önerilen yaklaşımların işletmelerin müşteri tercihlerine ve duygularına ilişkin kararlar alma konusundaki önemli potansiyelini ve pratik değerini ortaya koymuştur.

**Anahtar Kelimeler :** Doğal dil işleme, Derin Öğrenme, Duygu analizi, Bağlamsal dil modelleri, Statik dil modelleri, ELMo, BERT, Word2Vec, GloVe, FastText, DFFNN, LSTM, BiLSTM, CNN.

**Bilim Kodu :** 92416

## ACKNOWLEDGMENT

First of all, I would like to give thanks to my advisor, Assist. Prof. Dr. Kürşat Mustafa KARAOĞLAN, for his great interest and assistance in preparation of this thesis.

## CONTENTS

## LIST OF FIGURES

**Page**

## LIST OF TABLES

**Page**

## ABBREVITIONS INDEX

### ABBREVITIONS

| | |
|---|---|
| Acc | : Accuracy |
| AI | : Artificial Intelligence |
| AUC | : Area under the ROC Curve |
| BERT | : Bidirectional Encoder Representations from Transformers |
| BiLSTM | : Bidirectional Long Short-Term Memory |
| BCE | : Binary Cross-Entropy |
| CCE | : Categorical Cross-Entropy |
| CLM | : Contextual Language Model |
| CNN | : Convolutional Neural Network |
| DFFNN | : Deep Feed Forward Neural Network |
| DL | : Deep Learning |
| ELMo | : Embeddings from Language Models |
| F1 | : F1 Score |
| GloVe | : Global Vectors for Word Representation |
| LSTM | : Long Short Term Language |
| MCC | : Matthews Correlation Coefficient |
| MSE | : Mean Squared Error |
| ML | : Machine Learning |
| MLP | : Multi-Layer Perceptron |
| NLP | : Natural Language Processing |
| P | : Precision |
| R | : Recall |
| ReLU | : Rectified Linear Unit |
| RNN | : Recurrent Neural Networks |
| ROC | : Receiver Operating Characteristic |
| SA | : Sentiment Analysis |

SLM : Static Language Model

# PART 1

# INTRODUCTION

Artificial Intelligence (AI) and Natural Language Processing (NLP) are critical to developing areas that change how we live and work, such as speech recognition and translation. In general, they are becoming heavily used in our daily lives.

Deep Learning (DL) is a subfield of Machine Learning (ML) done through artificial neural networks with many layers and is also known as deep neural networks [1]. These DL algorithms are designed to automatically learn from data and improve their performance over time without explicit programming. Computer Vision (CV) is one of the most popular applications of DL, as it is used to develop algorithms for image and video recognition, such as recognizing objects, faces, and even emotions. It is also widely used in sophisticated applications such as autonomous vehicle recognition technology [2,3].

Another primary application of DL is its use in the field of NLP, such as its use to develop algorithms for Sentiment Analysis (SA) [4], question answering [5,6], and entity recognition projects [7,8]. Other existing applications include speech recognition, where DL is used to create algorithms to transcribe and translate spoken language into text. This technology is used in dictation programs and voice-activated virtual assistants, such as Siri and Alexa, allowing users to enter text simply by speaking. The importance and power of DL lies in its ability to automatically learn from data and improve its performance over time without having to use explicit programming. It makes it a powerful tool for solving complex problems in various fields, from CV and NLP to finance and healthcare.

One of the most important applications of NLP is Sentiment Analysis (SA), such as SA of customer reviews, which is a technique that includes analyzing customer opinions to take a look at their views and feelings about a specific product or service, as it is an essential function for companies and institutions.

In other words, SA of customer reviews is an important task that can provide valuable insights into consumer opinions and attitudes, allowing companies to make informed decisions to drive growth and success.

**In this master's thesis**, the definition and applications of AI and NLP technologies and their importance in today's society will be explored. Also, the role of customer sentiment analysis in understanding consumer behavior and improving customer satisfaction will be examined. A comprehensive literature review will identify the most important applications of AI and NLP in SA.

This thesis aims to provide a comprehensive overview of AI, NLP, and customer SA and their importance in modern society. DL algorithms will achieve this through in-depth literature analysis, particularly using Static and Contextual Language Models to create semantic vectors. Static Language Models (SLMs), such as Word2vec, and Contextual Language Models (CLMs), such as Embeddings from Language Models (ELMo).

## 1.1. PROBLEM STATEMENT OF THIS THESIS

Because the customer and his satisfaction are the essences of shopping and the importance of electronic commerce in our life and its widespread dependence of many companies on selling their products online and because of the widespread online reviews and the importance of knowing the feelings hidden within them for what can benefit the development of the company and improve its products and find solutions and satisfy customers, and because of the difficulty of classifying and studying them manually It is necessary to develop an automated system that analyzes the sentiments embedded in the customers' comments based on the customer's evaluations attached to the comments.

## 1.2. MAIN OBJECTIVES OF THIS THESIS

- Introducing a novel approach to SA: The study proposes a novel approach using Deep Learning-based methodologies that utilize static and contextual language models. This approach is evaluated on the Amazon dataset, a publicly available benchmark dataset for SA.
- Comparison between the performance of both SLMs and CLMs: This study compares in terms of performance between SLMs as FastText and CLMs as ELMo, and this comparison will be using special measures of deep learning models to determine the most efficient model in SA.
- Development of several DL models for SA: The study proposes several DL models, namely convolutional neural network (CNN), Long Short Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM) and deep Feed Forward. These networks receive a contextual or static sequential representation of a given text as input. These networks were trained to predict the type of sentiment in a text.
- Reaching High Accuracy for SA: The proposed approach performs SA with high accuracy across the Amazon dataset.
- Providing insights for decision-making: Given that SA is essential, this approach provides critical information for decision-making for many areas, such as social media monitoring, marketing, and brand reputation monitoring.

## 1.3. ORGANIZATION OF THE THESIS

The organization of this thesis is general; it consists of five main sections: introduction, literature search, background theoretical, methodology, and finally, summary containing results with discussion, conclusions and feature studies.

In the first part of the thesis, the information summarizing the suggested approaches and their importance is presented under the title of "Introduction", together with the aims and contributions of the thesis study. The second part of the thesis is devoted to literature research. In the third part, we will talk about the background theory of our project involves language models, DL models with evaluation and measurement criteria, and SA and NLP importance. In the fourth part, we will explain our project methodology. In the fifth part, we will study our results, analyze them, and compare the performance of the language and DL models; then, we will discuss future works and a summary of our work.

# PART 2

## LITERATURE REVIEW

According to the literature, there are several studies conducted in the field of sentiment analysis, in which the methods of analysis varied, but it has been noted that most of them fall into these types of systems: systems that use lexicon methods and machine learning methods with tow kind of learning (supervised and unsupervised) [9]. Also, the SA that was mentioned in the literature in general falls into the following categories: SA of the text [10], SA of the document [11], and Aspect-based SA [12]. On the other hand, according to previous studies, the types of data for which sentiments were analyzed were observed. Some studies analyzed sentiments for texts, and the other analyzed sentiments for audio clips. Among the studies, sentiments were analyzed for video clips as well as for images.

SA studies are widely employed in numerous fields, including political [13,14] analysis and social media tracking [15]. Their popularity has grown in recent years. In the literature, studies to identify sentiment in social media are also presented [16,17]. In the marketing sector, some studies have also shown evidence of sentiment in consumer reviews or comments [18]. Since many researchers studied how people felt and reacted to this illness at the time of the Corona epidemic, SA also had a significant role in this period [19,20].

In the healthcare domain, researchers have explored the potential of SA to identify patient satisfaction and dissatisfaction with healthcare services. For example, a study used SA to analyze online reviews of healthcare providers and found that SA can provide valuable insights into patient experiences and preferences [21].

Several studies in the field of sentiment analysis of customers' opinions of what is important in marketing and e-commerce showed high results and performance. Amazon Data was also very suitable for these studies.

The Amazon dataset used in this study contains a diverse range of data. As a result, this dataset is a great resource for academics evaluating model performance. The literature [22] describes a hybrid strategy for performing SA tasks on the Amazon dataset that combines SVM and k-Means. Another research project [23] uses Amazon dataset mobile phone reviews to execute SA problems with the Bidirectional Encoder Representations from Transformers (BERT) as language model. A similar study [24] used a CNN-based SA technique in text representation with Word2Vec for SA on cell phone reviews. Other research [25] that uses TF-IDF for feature extraction and LSTM as a classifier model achieves effective performance results on the Amazon dataset.

It is also important to note that different groups used different labelling methods when dealing with the dataset in question. For example, in one study [23], comments with one and two stars were considered unsatisfied, whereas three stars were defined as neutral, and four and five stars were labelled as satisfied. Another study [26],on the other hand, converted the classification problem into a binary issue by labelling reviews with (1 and 2) stars as negative and (4 and 5) stars as positive, primarily removing the 3-star ratings. That simpler binary classification outperformed the more sophisticated task of categorizing the sentiment into multiple categories used in our study.

Furthermore, many researchers also investigated the SA for customer opinions from several sources. For example, in this study [27], researchers used the "Chat" application to collect customer comments over three years, as these comments were rated from one to five stars, with ratings of one and two indicating negative reviews, three indicating neutral feedback, and four and five indicating positive reviews.

The great role of machine learning in text classification and sentiment analysis has been noted according to the literature, for example, Random Forest [28] or SVM [29,30] can be used, and there are several experiments that relied on linguistic analysis and proved their ability to obtain good results in the field of understanding the context of sentences and words and classifying texts [31]. Besides ML, It has been observed that emotions can be detected in the text using several models such as LSTM is a neural network developed on the basis of Recurrent Neural Networks (RNN) as a solution to the vanishing gradient problem that increases the complexity of training these data [32,33]. There is also CNN model [34,35]. In addition, there is a study based on the transformer model to apply the SA task. For example, emotions are classified using FastText or BERT as a transformer-based deep learning model. It proved the ability to get high classification accuracy of up to 88% [36,37] .

In terms of representing words as an important step in all text classification or recognition projects, a great diversity has been observed to achieve this thing, some of which were within the SLMs and the other within the CLMs, in the SLMs, such as relying on FastText to represent words as a well-known language model [38] and word2vec as well [39], and in the CLMs, which expresses the representation of words and sentences depending on the context in which it came, such as RoBERTa [40] and BERT (Pre-trained Generative Transformer) [41] it relies on its work on techniques of Deep Learning to creates vectors by analyzing text to take into account the words that come after and before the word. In addition to that, there are also studies and researches that have combined several models to achieve word representation, for example, merging GLOV and FastText as two SLM models and resulting in a single representation of the word so that each word in the text will have one vector, and also showed good and high results from this integration or merge [42].

However, not much work has been done on sentiment analyses of customer reviews based more techniques, customer retting and using multi classification. This thesis will be tested on a comparable data set based on the literature and performance evaluations of the proposed approaches will be carried out.

## PART 3

## THEORETICAL BACKGROUND

### 3.1. NATURAL LANGUAGE PROCESSING

A subfield of AI known as Natural Language Processing (NLP) is concerned with using natural language in the interactions between computers and people. To perform tasks like language translations, sentiment analysis, and text summarization, NLP tries to make computers understand and realize human language [43]. The relationship between AI, SA, and NLP is shown in Figure 3.1.

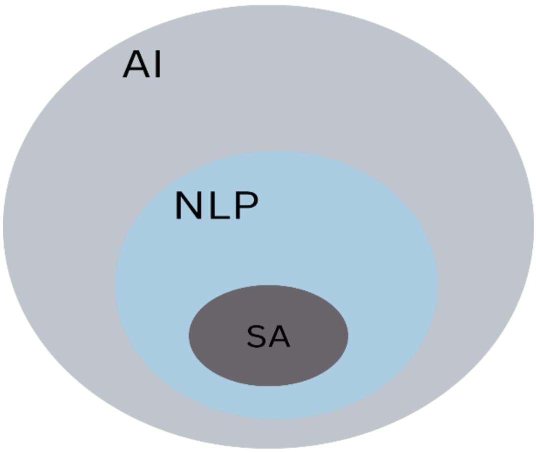


Figure 03.01. Relationship between AI,  SA, and NLP.

Due to the huge amounts of mixed unformatted data produced by social media, online reviews, and other kinds of digital communication, NLP has played a huge role in real life, this data can be text, audio or video, the type of data that NLP can process and the famous applications of NLP are shown in Figure 3.2. Ideas from this data can be used by businesses and organizations to improve customer services, evaluate customer sentiment and feelings, and discover new ways.

A wide range of industries, including business and healthcare or education, entertainment, and more, are using NLP. We'll see some of the most important NLP applications as follows:

Text Classification: NLP classifies text into several groups or themes. Many text data, such as news, social media posts, and customer reviews, can be organized and classified in this way. Spam detection, sentiment analysis, opinion mining [44], and content recommendation systems can all benefit from text classification.

Machine translation: NLP creates machine translation systems that can convert texts between multiple languages. Different methods will be used to translate the text as accurately as possible, just as statistical machine translation, rule-based machine translation, and neural machine translation. Online content translation, global corporate communications, and language learning are some of the many uses of machine translation [45].

Speech Recognition: Speech recognition systems that translate spoken language into written text are created using NLP. These systems use many methods, just as phonetic modelling, language modelling, and machine learning. There are many uses for speech recognition, including language learning, virtual assistance, and dictation software [46].

Named Entity Recognition: Named entities, simply as names, places or organizations, and dates, are being recognized and extracted from the texts using NLP. Applications for this include sentiment analysis, content recommendation systems, and also search engines [8].

Question-Answering: NLP is used to create systems that can answer questions and queries in the natural language precisely. These systems use various methods, just as information retrieval, semantic parsing, and machine learning, to understand and comprehend the purpose of these questions to deliver precise responses. There are many uses for question-answering, such as chatbots, virtual assistants, and customer services [5].

Text summarizing: Text summarizing systems that can provide summaries of very long texts or articles are developed by using the NLP. These systems will use many technologies to produce briefs that are accurate, short, and understandable, such as extractive summarizing, abstractive summarization, and DL Research paper summary is one of the many uses for text summarizing, which also has a lot of applications in news gathering and content recommendation systems [47].

In conclusion, NLP has so many uses in a lot of many different fields. Those tools can be helpful in understanding in a better way and the use of huge amounts of text data by researchers and individuals, as well as making it more effective and efficient for communication systems.

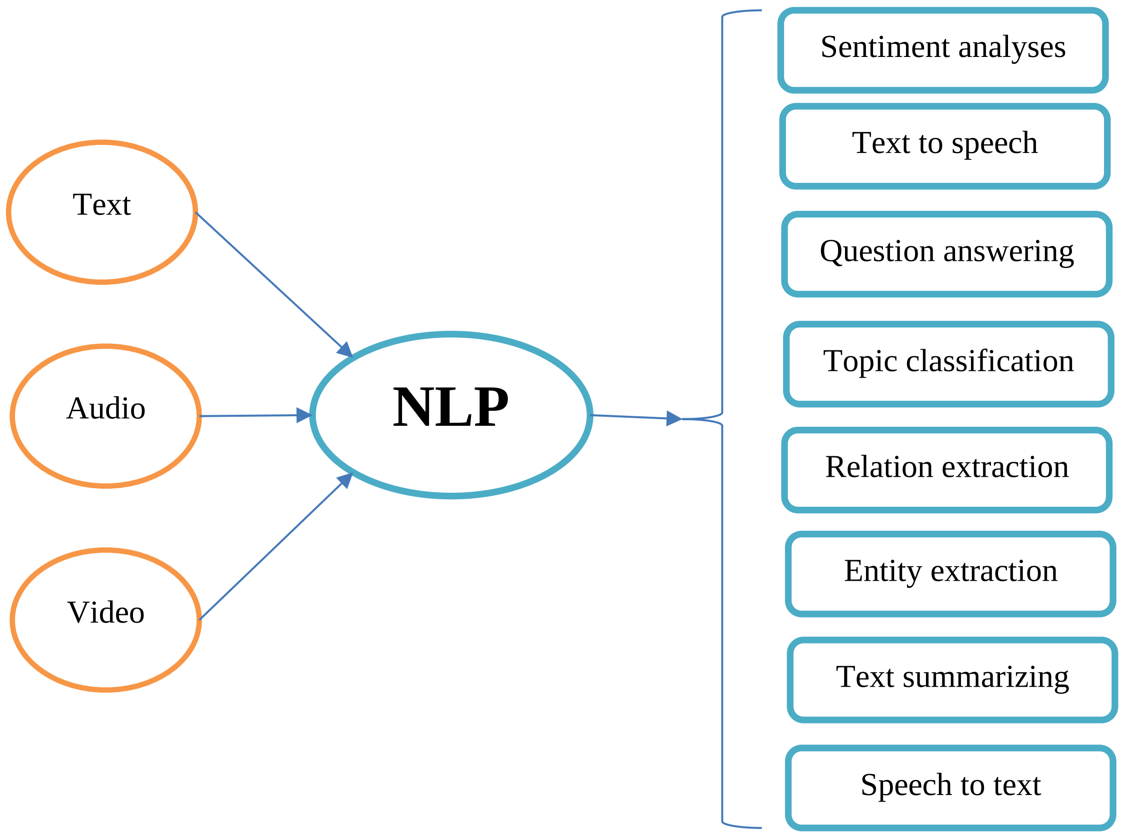


Figure 3.2. The types of data that NLP handle and a widely used applications.

### 3.2. SENTIMENT ANALYSIS

SA is the process of knowing the emotional expression of any text, image, audio, or video, such as detecting the feeling in the text that carries a positive or negative feeling, such as analyzing the feelings of a tweet or a customer review, using NLP with machine learning algorithms [48]. It is also called the opinion mining process [49], and SA is considered a technology of one of the text classification projects. The main applications of SA:

- Business-related management: The competition in the global co-op is intense due to the ever-changing market conditions. Everyone wants to make an advanced product that will make their customers happy. Organizations can collect requirements from all users and improve product efficiency through customer feedback to increase the value of their products.

- Decision Support: Building a website capable of making decisions is essential. The benefits of analysis include generating new insights that can help us make decisions in our daily lives, such as choosing the best restaurant for dinner, buying a new car, choosing a good movie to watch, and so on.

- Predictions and Trend Analysis: Monitoring public opinion using sentiment analysis allows anyone to predict market conditions, which is useful for trading and market sounding. It enables users to predict market trends [9].

The main types of SA as the following:

- SA of text: It analyzes the feelings of the sentences that make up the text.
- SA of document: It analyzes the feelings of the files' texts.
- Aspect-based SA: It is the analysis of the feelings of the text about more than one aspect or side [50]. An example of this type:

S1: "This device performs beautifully and is very powerful, but its color is undesirable."

S2: "This country is beautiful and has amazing scenery, but its people are bad."

The classification of feelings for the first sentence, S1, is positive in terms of performance and negative in terms of color. For the second sentence, S2, it is positive in terms of the beauty of the country and negative in terms of its people. The categories of SA are shown in Figure 3.3.

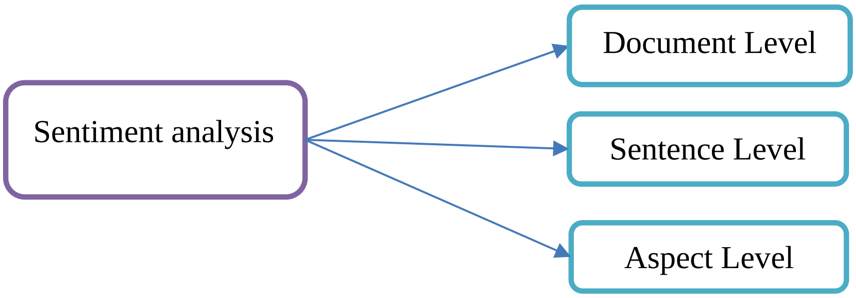


Figure 3.3. The main categories of SA .

### 3.3. DATA MINING

Data Mining (DM) is defined as the process of analyzing a quantity of data (usually a large amount) to find a logical relationship that summarizes the data in a new way that is understandable and useful to the data owner; that is, it is a technique that aims to infer knowledge from huge amounts of data [51].

DM and SA are two closely related disciplines that can be combined to gain insights from large textual datasets [52]. For example, by analyzing customer sentiment and feedback, companies can better understand their customers to make decisions to improve their products and services. The relationship between AI, NLP, SA, and DM is shown in Figure 3.4.

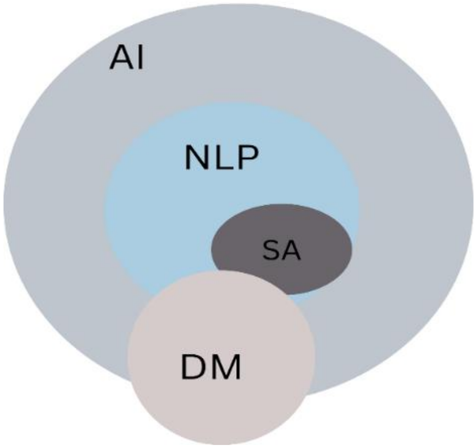


Figure 03.4. Relationship between AI, NLP,  SA, and DM.

### 3.4. DEEP LEARNING

DL is a method in AI that teaches computers to process data in a way inspired by the human brain. It is a part of machine learning and is a neural network composed of many neural layers to learn from and extract information from vast amounts of data.

DL models recognize complex patterns in sounds, text, images, and other data to uncover critical information and predictions. DL is becoming increasingly important in everyday life and implementing most NLP applications. The relationship between AI, NLP, SA, DM, ML, and DL is shown in Figure 3.5.

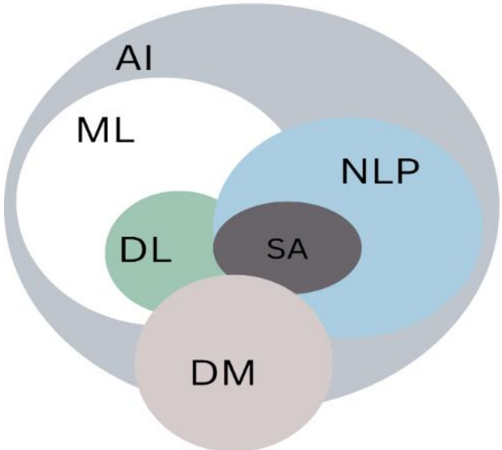


Figure 03.5. Relationship between AI, NLP, SA, DM, ML, and DL.

### 3.4.1. Supervised Learning

Supervised learning uses examples of known correct answers to train the network, and this means that the values we want to predict are known in our dataset. Through this, we want to build a model to build a relationship between input and output [53]. The supervised learning principle is shown in Figure 3.6.

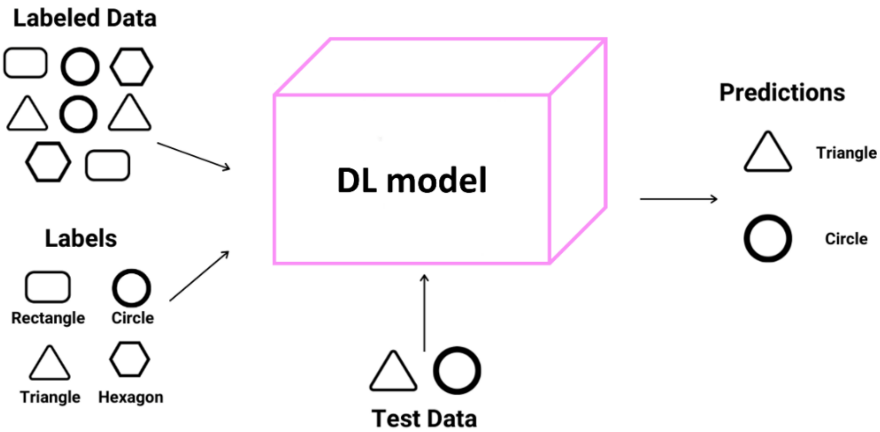


Figure 3.6. Supervised learning principle in DL.

### 3.4.2. Unsupervised Learning

Unsupervised learning applies when you have a data set but no labels. Unsupervised learning uses combinations of inputs and attempts to find patterns in the data [54]. For example, organize it into groups (Clusters) or find outliers (External Detection). The unsupervised learning principle is shown in Figure 3.7.

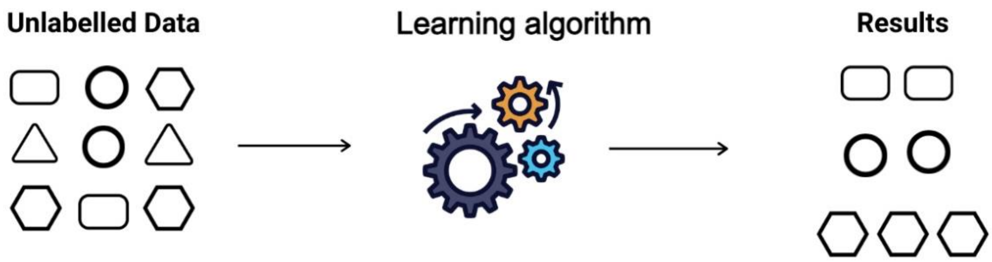


Figure 03.7. Unsupervised learning principle in DL.

### 3.4.3. Deep Feed Forward Network

Deep Feed Forward Neural Network (DFFNN) is an artificial neural network, or more specifically, a Multi-Layer Perceptron (MLP), the most widely used type of neural network. A DFFNN consists of an input layer, an output layer, and one or more hidden layers. Each layer of the DFFNN contains one or more neurons directly connected to the neurons of the previous layer and the next layer. It is called feed forward because each neuron is connected to all neurons in the next layer, and each connection has a weight value [55,56]. Figure 3.8 shows DFFNN with three inputs, two outputs, and a hidden layer containing five neurons.

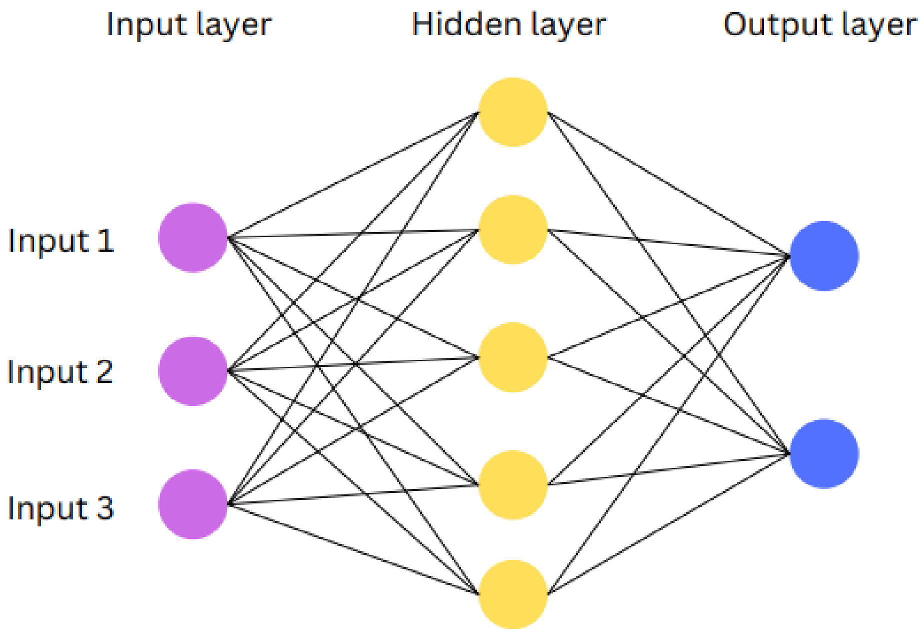


Figure 03.8. The architecture of DFFNN consists of 3 layers.

Most neurons in the DFFNN are similar. Each has many input links (it takes the output values of several neurons in the previous layer as input) and many output links (the response passes through many neurons in the next layer). The values retrieved from the previous layer are added with certain weights, and each neuron adds a bias term. The activation function transforms this sum, and the activation function F may differ for different layers. That is, the output of each layer l with the input coming from layer l-1:

$$Yl = F\left(\sum_{i=0}^{n} Xi * Wi + b\right) \tag{3.1}$$

Where *n* is the number of neurons in the layer of l, *X* is the input of the layer of l, *W* is the weights, *b* is the bias value, and *F* is the activation function of this layer.
The cost functions of training are the mean square error when the task is regulation and cross-entropy when it is classification. These models are called feed forward because information flows from the input to the output layer, where the connections are forward and do not form cycles (as in the case of recurrent networks) [57,58].

### 3.4.4. Long Short-Term Memory

LSTM is a neural network architecture on a wide range that is used in deep learning for processing and dealing with sequential data. Also it includes NLP natural language processing and speech recognition too. LSTMs belong to RNNs, meaning the category of frequent or recurrent neural networks is also designed specifically to deal with vanishing gradients in RNNs. The main topic is to deal with traditional RNNs' limitations and maintain enhanced results [59].

Deeply, LSTM concludes a memory cell, allowing the network to forget or retrain information selected from previous time steps. This would be achieved by gates controlling the information's flow in and out of the memory cell.

In the training phase, these gates would be learned and adjusted according to input and the current state of the networks. Often, there would be confusion between "Cell State" and "Hidden State" in LSTMs. The two components would serve purposes separately. The cell state looks to encode gathered data from previously processed time steps, while the hidden state would differentiate data from a previous time step.

Two types of normalization equations are generally used In LSTM: the sigmoid function (lower-case sigma) and the (tanh) function of the hyperbolic tangent. This intentional choice would offer a conjectural explanation for the application.
LSTM network fundamental elements include:

- Input gate: Specifies the importance of information in the current time step, which would be stored in the memory cell.
- Forget gate: Specifies which information would be neglected from the previous time step and not stored in the memory cell.
- Output gate: Specifies which information would be inherited from the memory cell to the next layer in the network.
- Memory cell: It is a long-term Memory component of the network which stores the information from previous time steps.

During the training phase, based on inputs and the network's current state, the LSTM learns to set and regulate the gates, allowing it to store or forget the information from previous time steps as it requires. This ability allows the LSTM to model long-term dependencies within sequence data effectively.

In addition, the gates which are used by the network strongly help in keeping the most significant information, so it would greatly improve the output's quality. This feature is significantly very valuable in these cases, like reading promotional content which contains information, prices and dates. In similar cases, pointless words like extraneous and prepositions terms are neglected, which is very similar to the function which LSTMs perform [34].

In summary, LSTMs are a special type of neural network designed specifically to handle a sequence kind of input within long-term capabilities. That is, it is good for sequence data. LSTMs can efficiently forget or remember any information from previous time steps by adding a memory cell and a set of gates for the information flow control. Because of their flexibility, LSTMs can effectively model complex data sequences, making them a common proper choice for many speech recognition applications and NLP [60].

### 3.4.5. Bidirectional Long Short-Term Memory

When we are dealing with a huge amount of data, the model must be aware of the relationship between the suffix and the antecedent of the word as well. To solve this problem, the bi-directional network was introduced. This network could be used binary with LSTM and RNN, where due to limitations in LSTM the input was presented in both directions, left to right and right to left. Additionally, this is not a back propagation. That's just the input that was provided by both sides. That is, it feed once from start to end to access past information and once from end to start to access future information [61,62].

### 3.4.6. Convolutional Neural Networks

CNN is a variant of a deep learning network, the main components of a CNN for text analysis:

**Embedding layers:** An embedding layer is a particular hidden layer in a neural network. In other words, this layer converts the input data from a high-dimensional space into a lower-dimensional space so that the network can quickly process the data and understand how the inputs connect. In other words, it is used to make it easier to learn embedding words.

**Convolutional layer:** The convolutional layer is used to extract features and patterns from the numeric representation of text. Convolutional nodes can explore possible associations between words and characters in a text. The convolution nodes are traversed through the numeric representation of the text and apply a convolution operation to calculate the raster dot between the numerical values and weights in the convolution kernel. The convolutional layer produces multiple feature maps that represent the response to the pattern in the numeric representation.

**Pooling layer:** This layer is used to reduce the spatial dimensions of the features extracted from the bypass layer, and this is done by grouping adjacent values into certain areas, such as Max Pooling which selects the largest value in the pooling area, the pooling layer helps reduce dimensions and enhance the salient features of the digital representation of text.

**Dropout:** Dropout is a technique frequently used in CNNs to prevent overfitting during training, this technology on each update, randomly sets a portion of activations to zero, which helps prevent the network from becoming too dependent on any given set of features, that is, this technique forces the network to learn more powerful and generalized features.

**Fully connected layer:** After convolution layers and pooling, duplicate text-connected layers can be added to the classification tasks. Fully connected layers consist of several interconnected neurons, which calculate the weight and bias of the extracted features and define the text class.

These components combine into a convolutional neural network for text analysis to extract critical information and relevant features from the input data and classify texts into different categories [63,64].

### 3.4.7. Hyperparameters Of Training In DL

They are settings that the user or the data scientist determines, and he can modify them to suit the model's performance. These settings control the behavior and performance of deep learning algorithms, as they have a major role in the final results of model training [65]. These parameters are not learned like the biases and weights of neural networks learned through optimization. Rather, the user predetermined them after understanding the problem and its type. This task is very important, as it is possible to change one of its values to avoid falling into learning problems, such as avoiding the occurrence of overfitting. Figure 3.9 shows the roll of hyperparameters in DL.

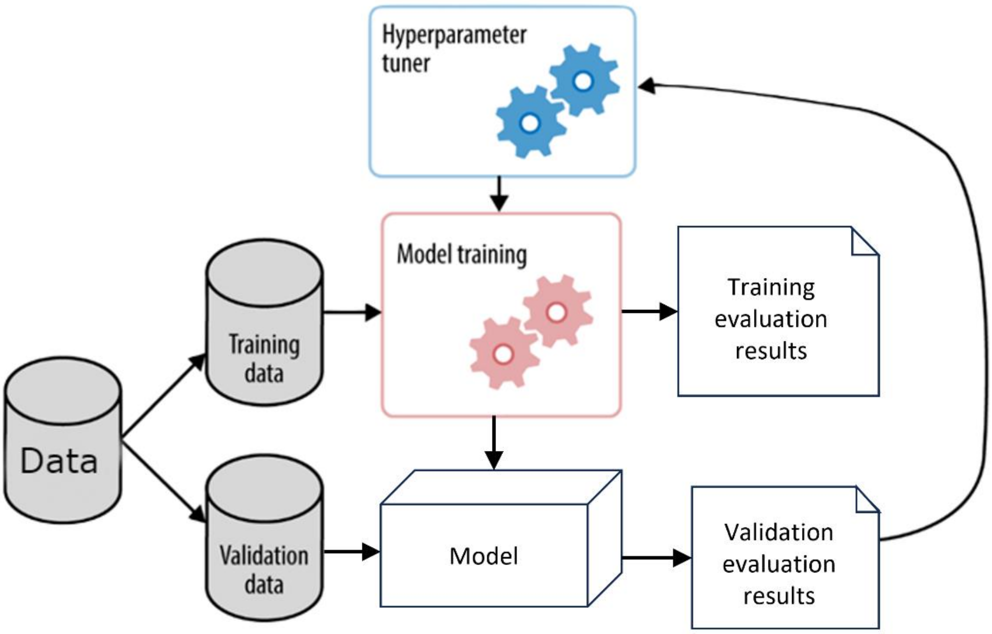


Figure 3.9. The roll of hyperparameters in DL.

The most famous of these parameters:

1. Loos function: It is also called the cost function, and it is a function that measures the value of the discrepancy between the actual output of the neural network and the correct expected output; in other words, the value of error and loss is determined during training, and it is a way to give a signal to the model to try to reduce this error by adjusting its parameters. The appropriate function is chosen according to the problem type, such as classification or regression. The most loos functions are:

- Mean Squared Error (MSE): It is a loss function that is widely used for regression tasks, as it calculates the average squared difference between the expected values and the real values, and the goal in training is to reduce this value, the mathematical equations is:

$$MSE = \frac{1}{n} \sum_{i=0}^{n} |y(i) - a|^2 \quad (3.2)$$

Where *a* is the predicted value or output vector, *y(i)* is the actual value of the input *i* (training example), *n* is the number of inputs for training, and *i* is the input.

- Binary Cross-Entropy (BCE): It is commonly used in binary classification problems, measuring the difference between the expected and real binary labels.
- Categorical Cross-Entropy (CCE): It is commonly used in multiclass classification problems, it calculates the difference between expected class probabilities and real class labels, and the goal is also to minimize this value, the mathematical equation is:

$$CCE = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij} * log\left(p_{ij}\right) \tag{3.3}$$

Where *m* is the number of classes and $y_{ij}$ is the actual output for input *i*, $p_{ij}$ is the predicted value of the input *i*, and n is the number of inputs. The BCE is the special case of Categorical Cross-Entropy when *m* = 2.

2. Activation function: It is a mathematical function that is applied to the output of a neuron. The most activation functions are:

- Rectified Linear Unit (ReLU): It is a commonly used function that outputs the same input if positive and zero; otherwise, its mathematical equation is:

$$f(x) = max(0, x) \tag{3.4}$$

ReLU is a standard function in many neural network models because of its simplicity, computation speed, and ability to mitigate the vanishing gradient problem.

- Softmax: They are commonly used in multiclass classification problems. It takes a vector of real values and then normalizes them into a probability distribution. That is, it assigns probabilities to each category. Its mathematical equation is.

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{3.5}$$

- Sigmoid: It maps the input to a value between 0 and 1 and thus provides smooth and limited activation. Its mathematical equation is:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.6}$$

3. Learning rate: The value that the weights are updated during training. It is usually between 0 and 1, as the learning rate when it is very small can cause the process to stop in the process, and when it is too large causes the model to converge faster to a suboptimal solution.
4. Number of epochs: The number of learning times on all data sets allocated for training.
5. Number of layers: The number of hidden layers or the depth of the neural network.
6. Number of neurons per layer: The number of neurons in each hidden layer where the number of neurons in each layer affects the model's power.
7. Batch size: It is a value specifying the number of training samples taken in one pass forward and backward. This value can affect convergence speed and memory requirements.

### 3.4.8. Evaluation Metrics Of DL Models

There must be measures that measure the performance and activity of the trained deep learning model to see if it has been properly and sufficiently trained or needs modification and retraining. Suppose these measurements gave a result that was not good or acceptable. In that case, we will know that we must repeat the training process by changing one of the parameters of the hyperparameter, such as removing or adding more hidden neural layers or changing the number of training epochs, or it is possible to change the entire neural network by choosing another type of network. Some of the most popular of these metrics are:

- Receiver Operating Characteristic (ROC): A curve expressing the true positive rate versus the false positive rate for different threshold values.
- Area under the ROC Curve (AUC): It is the value of the area under the ROC curve. A value of 1.0 indicates ideal classification performance.
- Matthews Correlation Coefficient (MCC): It is used primarily in cases where classes are unbalanced; it ranges from -1 to 1, where a value of 1 indicates the best performance.
- Learning curves: They are graphs that show how much the model has learned and trained during the training epochs while training the model, and this helps to determine whether the model and the hyperparameters are appropriate or not.
- Accuracy (Acc): It is the ratio of the number of correct classifications for all types of classes over the total number of samples, i.e., it expresses the ratio of the correct classification of the entire data, and it is one of the well-known criteria in classification tasks. Its mathematical formula is in Equation 3.7.
- Recall (R): It is also called true positive rate or sensitive. It is a measure that quantifies the model's ability to identify all positive cases. It is the ratio of correct positive ratings over the sum of the correct positive and false negative ratings. Its mathematical formula is in Equation 3.8.
- Precision (P): It measures the model's ability to identify positive cases correctly and is the ratio of true positives to the sum of true and false positives. Its mathematical formula is in Equation 3.9.
- F1 Score (F1): A balanced measure of model performance by considering accuracy and recall. Its mathematical formula is in Equation 3.10.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.7}$$

$$R = \frac{TP}{TP + FN} \tag{3.8}$$

$$P = \frac{TP}{TP + FP} \tag{3.9}$$

$$F1 = 2 * \frac{(P * R)}{(P + R)} \tag{3.10}$$

While *TP* means "True Positives," *TN* as "True Negatives," *FP* as "False Positives," and *FN* as "False Negatives" [66] .

- Confusion matrix: It is a table that expresses the performance of the deep learning model for the classification of each class, as it shows the number of true positive predictions, true negative, false positive, and false negative predictions.

## 3.5. PRE-PROCESSEING THE TEXT DATA

Textual data processing in NLP is the first step in cleaning and transforming raw textual data to make it suitable for subsequent analysis and modeling. Data processing is important in NLP because it helps remove unnecessary information, normalize text, and increase the effectiveness of the following algorithms and models [67]. Figure 3.10 shows an example of a review after pre-processing. The main steps of pre-processing:

**Cleaning texts for pre-processing:** The following components were considered for this process.

- All marks, punctuation, emoji, html tags, and numbers, that is, the removal of all symbols that do not represent letters of the alphabet.
- Null values: The dataset may contain records that are null values, and therefore, it is necessary to remove them because they are hindered in the analysis, so we removed all null values from the data.
- Multiple spaces.

**Contraction replacement**: Data may contain an apostrophe used for abbreviations. Ex: “isn’t” for “is not.” This can change the meaning of a sentence or word. So it is better to replace these apostrophes with standard dictionaries. Creating a dictionary consisting of the actual value and the contraction is possible.

- The contraction: n't --> not, ’ll --> will, ‘s --> is, 'd --> would, ‘m --> am, 've --> have, ’re --> are.
- And this is the dictionary: Apos_dict={" 's":" is "," n't":" not"," 'm":" am"," 'll":" will", " 'd":" would"," 've":" have"," 're":" are"}

**Normalization:** Converting all text to one case, like converting all words to lowercase, to avoid treating the same words as different due to case changes, sometimes lowercase and uppercase.

**Stop word removal:** Remove commonly used words that only carry a little semantic meaning (called stop words), such as articles, prepositions, and conjunctions that may confuse the data without adding significant value. The stop words in the English language are:

[‘i’, ‘me’, ‘my’, ‘myself’, ‘we’, ‘our’, ‘ours’, ‘ourselves’, ‘you’, “you’re”, “you’ve”, “you’ll”, “you’d”, ‘your’, ‘yours’, ‘yourself’, ‘yourselves’, ‘he’, ‘him’, ‘his’, ‘himself’, ‘she’, “she’s”, ‘her’, ‘hers’, ‘herself’, ‘it’, “it’s”, ‘its’, ‘itself’, ‘they’, ‘them’, ‘their’, ‘theirs’, ‘themselves’, ‘what’, ‘which’, ‘who’, ‘whom’, ‘this’, ‘that’, “that’ll”, ‘these’, ‘those’, ‘am’, ‘is’, ‘are’, ‘was’, ‘were’, ‘be’, ‘been’, ‘being’, ‘have’, ‘has’, ‘had’, ‘having’, ‘do’, ‘does’, ‘did’, ‘doing’, ‘a’, ‘an’, ‘the’, ‘and’, ‘but’, ‘if’, ‘or’, ‘because’, ‘as’, ‘until’, ‘while’, ‘of’, ‘at’, ‘by’, ‘for’, ‘with’, ‘about’, ‘against’, ‘between’, ‘into’, ‘through’, ‘during’, ‘before’, ‘after’, ‘above’, ‘below’, ‘to’, ‘from’, ‘up’, ‘down’, ‘in’, ‘out’, ‘on’, ‘off’, ‘over’, ‘under’, ‘again’, ‘further’, ‘then’, ‘once’, ‘here’, ‘there’, ‘when’, ‘where’, ‘why’, ‘how’, ‘all’, ‘any’, ‘both’, ‘each’, ‘few’, ‘more’, ‘most’, ‘other’, ‘some’, ‘such’, ‘no’, ‘nor’, ‘not’, ‘only’, ‘own’, ‘same’, ‘so’, ‘than’, ‘too’, ‘very’, ‘s’, ‘t’, ‘can’, ‘will’, ‘just’, ‘don’, “don’t”, ‘should’, “should’ve”, ‘now’, ‘d’, ‘ll’, ‘m’, ‘o’, ‘re’, ‘ve’, ‘y’, ‘ain’, ‘aren’, “aren’t”, ‘couldn’, “couldn’t”, ‘didn’, “didn’t”, ‘doesn’, “doesn’t”, ‘hadn’, “hadn’t”, ‘hasn’, “hasn’t”, ‘haven’,

“haven’t”, ‘isn’, “isn’t”, ‘ma’, ‘mightn’, “mightn’t”, ‘mustn’, “mustn’t”, ‘needn’, “needn’t”, ‘shan’, “shan’t”, ‘shouldn’, “shouldn’t”, ‘wasn’, “wasn’t”, ‘weren’, “weren’t”, ‘won’, “won’t”, ‘wouldn’, “wouldn’t”].

**Lemmatization of data:** Lemmatization is similar to stemming in his work but with a much better quality than stemming. Correct words are always produced in lemmatization because after cutting the suffix from the word, the resulting word is always meaningful and belongs in the dictionary.

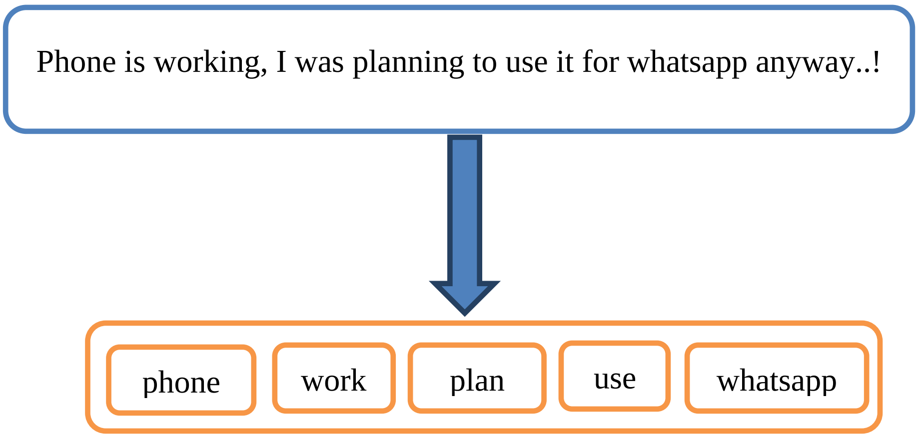


Figure 03.10. Example of review after pre-processing

**Tokenization of data:** It is necessary to represent words with numbers because deep learning or machine learning algorithms do not deal with words but with numbers. That is done by applying the following:

- A dictionary is created for all the words in the dataset so that each unique word in the data set will be the key, the number will be the value, and no two words have the same number. Each word refers to a unique number, and the numbers will start from a value equal to 1 because zero represents the padding value.

Eg:

"The man sat on the chair.” the dictionary is: word_index["the"] = 1, word_index["man"] = 2,
word_index["sat"] = 3; word_index["on"] = 4, word_index["chair"] = 5.

- Create vectors of numbers for all records in the dataset: where each word is replaced by its value in the dictionary.

Eg:

"The man sat on the chair." ➔ [1, 2, 3, 4, 1, 5]

**Padding:** It is carrying out the process of padding the vectors to a specific number equal to or greater than the longest dimension of the vector so that they are all of the same length.

### 3.6. CROSS VALIDATION

This technique divides the data into k folds, where n is a number greater or equal to 3, one for testing and the rest for training. For example, if the value of k is 5, the data will be divided into five folds, one for testing and 4 for training. The training and testing process will be carried out in k iterations; in each iteration, the test and training set will be different. Thus, we guarantee training on and testing the entire data [68]. Figure 3.11 illustrates the principle of cross-validation in training when k is 5.

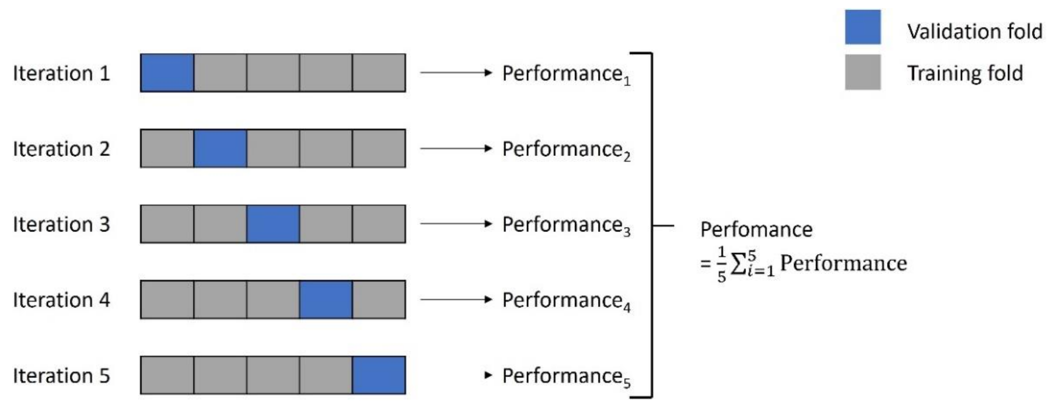


Figure 03.11. The princple of cross validation in training when k is 5.

The performance of the model will be calculated in each iteration. After the end of iterations, the final performance will be calculated by calculating the arithmetic average value of the model's performance in each iteration.

## 3.7. LANGUAGE MODELS

The language model is a type of AI model, a pre-trained model, a statistical model that estimates the probability distribution of words or characters in a sequence that enables computers to understand and generate human-like language and plays an important role in NLP applications. Its main goal is to predict the next word in a series of words by training on a large corpus of text and then analyzing it. Some of these neural language models have made great progress in capturing context and creating coherent text [69].

### 3.7.1. Static Language Models

They are traditional models, also called non-contextual models. These models rely on fixed representations of different words or symbols. In addition, these models specify probabilities for words or sequences of words without taking into account the surrounding context of the word.

#### 3.7.1.1. Word2Vec

It uses an unsupervised learning approach and a shallow neural network trained on word embedding. A text string is used as the input, and several strings of text properties are produced. The Word2Vec tool's primary function is to create matrices of related, identical, and similar words based on the mathematical similarity of each word. By evaluating each word's importance and value in the sentence, it also attempts to infer the meaning of the last word. Additionally, it learns whether a word is singular or plural, making it simple to finish the syntax of texts in the future. Along similar lines, the semantic closeness of terms is also highlighted. Mathematical expressions of word semantic similarity are available.

The Continuous Bag Of Words (CBOW) and Skip-gram are the two main word2vec techniques. The primary objective in CBOW is to produce a single word once the window size, which specifies the number of words before and after the needed word, has been established. Conversely, Skip-gram operates in contrast to

CBOW, which produces numerous words from a single input. In this case, the window size indicates the number of words before and after the specified single word [70,71]. Figure 3.12 shows the deferent between the way CBOW and Skip-gram work.

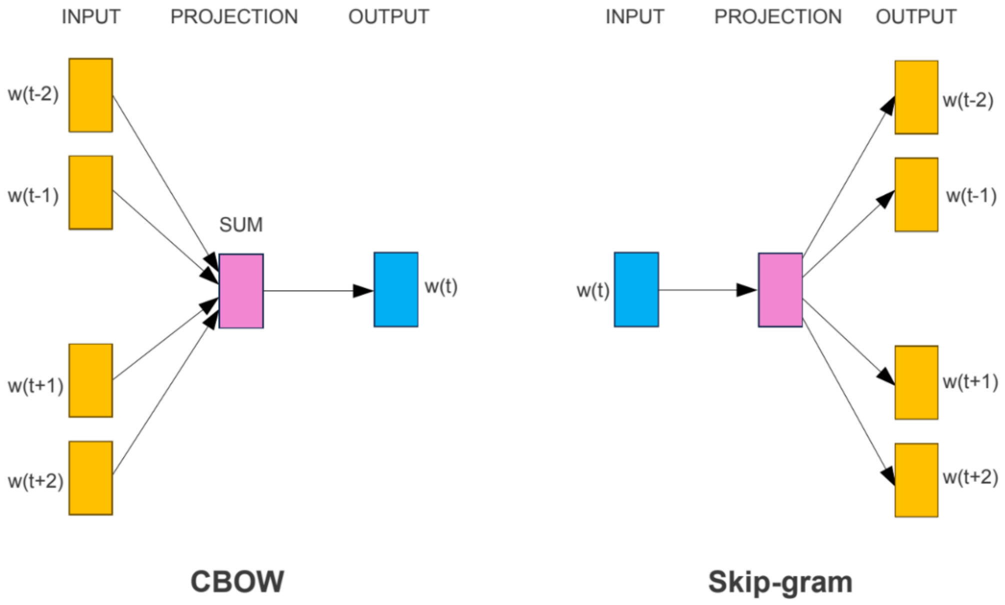


Figure 03.12. The deferent between the way CBOW and Skip-gram work.

### 3.7.1.2. FastText

It is a library that serves natural language processing applications such as big text representation, classification, and word embedding. This open-source library was developed by the Facebook team of researchers in the field of artificial intelligence. FastText is based on technology CBOW. And it is a kind of supervised learning method. Where the model learns according to the context and predicts the missing word closest to the meaning as output by using Hierarchical softmax after giving it several words as input, it is noteworthy that the Hierarchical softmax is also used for speeding the training and prediction.

In addition, FastText relies on Character n-grams, so it splits a single word into several words that are fragmented from the original word, where n-gram is a grouping of n consecutive text elements, which could be made up of words, symbols, or numbers, in other words, an n-gram is a method for handling sentences that contain words or words that contain letters. Where "N" stands for numbers and "grams" for words or letters [72,73]. For example, if we consider N=3 and we have <beautiful> word, then the sub-words will be as <be, eau, aut, uti, tif, ifu, ful, ul>.

#### 3.7.1.3. Global Vectors For Word Representation

Global Vectors for Word Representation (GloVe) is pre-trained concept is based on the ratio's probability and the Co-Occurrence matrix. After removing duplication, all of the terms in the text are included in the square co-occurrence matrix (rows number equals to columns number). The extent to which they coexist with other words in each context or sentence is shown by the common values in the table. It stands for Global Vectors of the Word Representation and is a form of unsupervised learning. GloVe debuted at Stanford University as an open-source project built on the concept of word representation as a vector to identify word context and semantic similarity. In order to construct the co-existence matrix, we must first decide the size of the window or the number of words that will be examined on the right and left of the window. For example, if the window is set to 2, we will determine each word's relationship to the two words that come before it and the two words that come after it. To assess the relationship between words in a corpus, conditional probability ratios can be calculated using GloVe vectors.

In other words, GloVe, which has been trained on a massive number of words to assess the semantic similarity of words to one another, is essentially the same as the word embedding approach [74].

### 3.7.2. Contextual Language Models

Contextual Language Models (CLMs) that consider surrounding context when estimating word probabilities. These models aim to create word embeddings or representations that capture a word's meaning and contextual information. They model leaders to deal with polysemy, resolve ambiguity, and generate responses more appropriate to the context. These models have gained popularity in recent years due to their superiority in tasks that require a deep understanding of context and context-dependent meanings. The downside is that they are computationally intensive and require large training data.

#### 3.7.2.1. Bidirectional Encoder Representations From Transformers

It is a language model developed by Google, it is considered a pre-trained model. It has improved NLP problem solving by understanding and analyzing the context of speech and the complex relationships between words for the sentences involved, BERT is a transformer model that leverages the power of unsupervised instruction to create high-quality word representations [75].

BERT is designed to form fixed-dimensional vectors called embedding or representation, so that each vector expresses the semantic meaning of the word and contextual information as well, this added a higher value to BERT's model, as it takes the full context of the word over the rest of the old language models that considered left-to-right or right-to-left contexts of the word.

**BERT Studies:**

It is a model trained on huge amounts of unlabeled text data. The training process consists of two main steps: pre-training and fine-tuning. In pre-training, it uses a technique called Persuasive Language Modeling (PLM) where it randomly hides some words in a sentence and then tries to predict them based on the remaining words, i.e. by context. This process allows BERT to gain a deeper understanding of the relationships between words in a sentence. In the fine-tuning step, which involves training BERT on specific NLP tasks, such as SA and answering questions, the model is fine-tuned to a target task-specific labeled dataset. For vector generation: BERT uses a multi-layer Transformer architecture comprising an encoder stack, each encoder layer consists of two sub-layers: a self-interest mechanism and a fully connected feed forward network.

Whereas, the self-attention mechanism is responsible for discovering contextual relationships between words by calculating the degree of attention between the target word and all other words in the sentence, that is, it measures the importance of each word in relation to the target word, it is calculated using the following equation:

$$attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) * V \tag{3.11}$$

Where: $Q$ is query matrix, $K$ is key matrix, and $V$ is valued matrix, whereas the softmax function adjust the attention score, also the square root for insure the attention scores are scaled properly [76, 77].

The output of the self-attention mechanism is fed into a feed forward network where it will apply a non-linear transformation to the representation of each word. This network consists of two layers with a ReLU activation function.

$$FFN(x) = \max(0, xW_1 + b_1)\, W_2 + b_2 \tag{3.12}$$

Where $x$ is the input vector of the word and *W1*, *b1*, *W2* and *b2* are parameters are parameters that are learned by the feed forward network.

By stacking multiple encoder layers, BERT is able to capture complex dependencies between words and generate rich word embeddings that capture both the meaning of the word and its contextual information.

**BERT_large_uncased:**

It is a specific type of the BERT model, which contains many parameters and is called large because it is a BERT with a large capacity to capture complex linguistic patterns, and it is called uncased because this model does not differentiate between lowercase and uppercase letters of words, it is also pre-trained on huge unlabeled text data.
BERT Large is distinguished by the ability to notice and capture contextual information and relationships between words in a sentence, and thus it will be able to create linguistic logic and form word embedding with high quality, the resulting vectors will all have length 1024.

It is crucially important in major challenges that require a deep understanding of the language, such as the tasks of answering questions, SA, and translation, because of its large capacity that enables it to deal more broadly with NLP tasks, as it gives more accurate results than the rest of the BERT models.

**BERT _base:**

$BERT_{base}$ would refer to a specific subset of the BERT model. "Base" means that it is a lesser BERT version with fewer parameters compared to larger subsets such as $BERT_{large}$. $BERT_{base}$ has been pre-trained on a large amount of unlabeled text data and has achieved to build high-quality word embeddings. It is very good at gathering contextual information and comprehending the relationships and meanings between words in the sentence.

Despite having fewer parameters, $BERT_{base}$ is still a sufficient model for many NLP tasks. It performs very well in tasks just as sentiment analysis, text classification, and named entity recognition. It can also be adjustable on specific kinds of downstream tasks to improve the performance and adapt it to the tasks.

The benefits of $BERT_{base}$ are in its effectiveness, as it takes less amount of computational resources and memory compared with larger subsets. It is convenient for applications where computing limitations are an issue or when the task does not need a comprehensive understanding of complicated language patterns.

In summary, $BERT_{base}$ is an eligible and commonly used language model that performs very well in a variety of NLP tasks. It achieves a balance between computing effectiveness and language comprehension, which makes it a popular NLP choice for a lot of practical applications [78,79,37]. Figure 3.13 shows the basic structure of $BERT_{large}$ and $BERT_{base}$ regarding decoders.

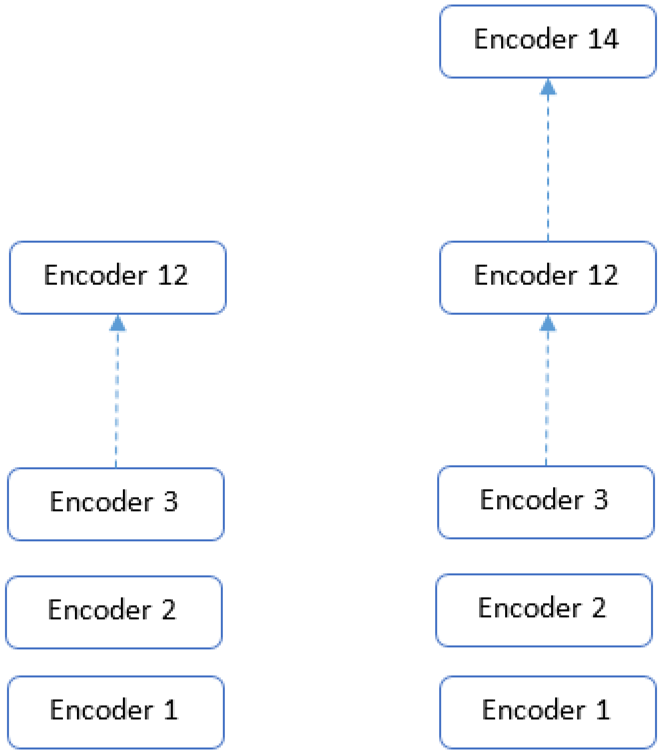


Figure 3.13. Basic architectures of $\text{BERT}_{\text{large}}$ and $\text{BERT}_{\text{base}}$.

#### 3.7.2.2. Embeddings From Language Models

ELMo is a deep contextual word representation model that captures both the meaning and context of words in a sentence by creating contextual word vectors, unlike traditional word embedding that provides static word representations. This model achieves this context by using a bidirectional language model based on a RNN. It consists of two layers: the character-level convolutional layer and the bidirectional LSTM layer.

The character-level convolutional layer produces word representations from the characters within the word by applying a set of convolutional filters to the characters and max-pooling to get a fixed-length representation of each word, thus capturing important information about the subwords.

The bidirectional LSTM layer takes word representations from the character level layer. This layer consists of two LSTM networks: one that processes the input sequence in the original order (Forward LSTM) and the other in reverse order (Back

LSTM), thus capturing the dependencies and relationships between words in both directions thus ensuring Complete understanding of the context. But to generate contextual word embeddings, ELMo uses a linear combination of hidden states from a bidirectional LSTM layer, and the weights are learned during the fine-tuning process. ELMo's strength lies in its ability to capture complex linguistic phenomena and polysemy, where the same word can have different meanings depending on the context, making it extremely effective for a wide range of NLP tasks [80,81].

Figure 3.14 shows the type of language models to create a semantic vector, with the example of SLMs and CLMs, and shows how the same word in CLM has more vectors depending on the context, whereas SLM takes one vector for all contexts.

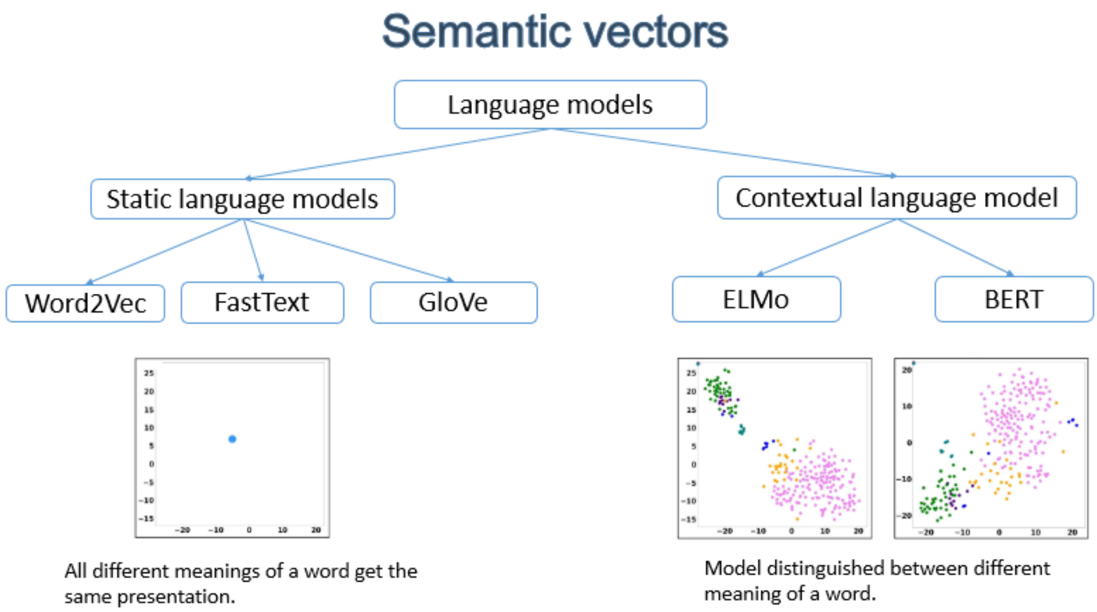


Figure 03.14. The different between SLM and CLM.

## PART 4

## METHODOLOGY

To achieve this system for analyzing sentiments for several categories was performed in the Python language and by using several software libraries such as Keras,
For the data used, Amazon data from the famous online store Amazon was depended on. It contains the customer's opinions of the products with another field that contains the customer's evaluation of the product as a number from one to five. Our dataset contains 50,000 customer reviews. Reviews with four or five stars are described as "satisfied", and reviews with one or two stars are labelled as "unsatisfied", whereas reviews with three stars are labelled as "neutral". Table 4.1 thoroughly breaks down the sentiment label distribution based on customer ratings.

Table 4.1. Statistical descriptive information about the dataset used.

| **Total number of reviews** | **50000** | | |
|---|---|---|---|
| **Shortest review** | 1 word | | |
| **Longest review** | 250 words | | |
| **Average word count** | 50 | | |
| **Label Number** | 3 | | |
| **Distribution of each class** | satisfied | neutral | unsatisfied |
| | 34000 | 4000 | 12000 |

### 4.1 PROPOSED APPROACHES

In this section, the basic stages of implementing the proposed system will be presented in detail, mentioning the values of the dimensions of the semantic vectors and the values of the hyperparameters for the training models. The architecture and components related to the execution of the approaches included in our study are presented in a way that explains the architecture in Figure 4. 4.

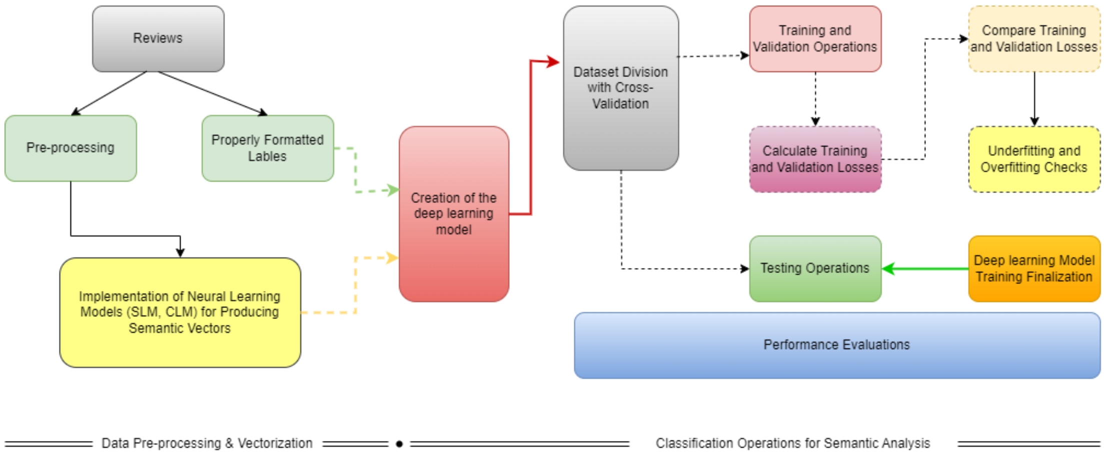


Figure 4.1. The architecture and components related to the approaches execution.

### 4.1.1. Pre-processing Stage

After selecting the data set, data pre-processing is the most critical step in all NLP projects. The programming libraries have been used to achieve some steps of preprocessed data, like Nltk and Kiras, where reviews have been preprocessed. The purpose of using each of them with the functions called from the libraries is shown in Table 6.

### 4.1.2. Language Model Applying Stage

After data pre-processing, data is represented both digitally and semantically by following several methods, including word embedding, i.e., representing each word in the dataset of a vector with x dimension, and the other is sentence embedding, i.e., representing each recode in our dataset in a vector, to achieve this we used SLMs and CLMs. An array of embeddings have been created, each line of which is a semantic vector. In the case of SLMs, each line of this array denotes a vector for a word in our dataset. According to the language model, it is an array with a dimension of x lines, the number of words mentioned in the dataset, and n columns. In the case of CLMs, each line of this array denotes a semantic vector for a sentence in our dataset.

According to the language model, it is an array of dimensions x, the number of sentences we have, and n columns.

The language models are used: Word2Vec, GloVe, and FastText as SLMs, and $\text{BERT}_{\text{base}}$ and ELMo as CLMs. Table 4.2 shows the details of each language model used in the study.

Table 04.2. The details of each language model used in the study.

| | **Dimension of vectors** | **Programming libraries** | **Language model type** |
|---|---|---|---|
| **Word2Vec** | 100 | Gensim | SLM |
| **Fasttext** | 100 | Gensim | SLM |
| **GloVe** | 50 | Glove | SLM |
| **$\text{BERT}_{\text{base}}$** | 728 | Flair | CLM |
| **ELMo** | 728 | Flair | CLM |

#### 4.1.3. The Classifier Model Creation And Training

After the vector representation of our dataset and getting semantic vectors, the data into two parts, training, and testing, was divided based on cross-validation using a k-fold of 5. Thus we have ensured that all the data has been trained and tested. Then DL models are created, DL models are:  deep feed forward model, CNN model, LSTM, and BiLSTM are created.

##### 4.1.3.1. Deep Feed Forward Model:

At the beginner is an embedding layer for all the models to become an income for the weights, which are the semantic vectors, then a flattened layer, one hidden layer with RelU activation function, which contains 100 neurons, and finally, output layer with softmax activation function and five neurons since we have 5 class. Table 4.3 shows the hyperparameter values for the training of the deep feed forward model.

Table 4.3. Hyperparameter values for training of deep feed forward model.

| Hyperparameters | Values | | | | |
|---|---|---|---|---|---|
| **Number of epochs** | BERT | ELMo | Word2Vec | GloVe | FastText |
| | 10 | 8 | 8 | 8 | 8 |
| **Batch size** | 256 | | | | |
| **Loss function** | Categorical Cross-entropy | | | | |
| **Activation functions** | Softmax | | | | |
| **Hidden layer no** | 1 | | | | |
| **Activation functions for hidden layer** | RelU | | | | |
| **Train Approach** | Cross-validation | | | | |
| **Optimizer** | Adam | | | | |

#### 4.1.3.2. CNN Model:

At the beginner is an embedding layer, a Convolutional layer with this parameter: filters is 50, kernel size is three and activation function is RelU. Convolutional layer with this parameter: filters is 100, kernel size is three and activation function is RelU. Convolutional layer with this parameter: filters is 200, kernel size is three and activation function is RelU. Then flatten the layer, two hidden layers with a RelU activation function, each containing 100 neurons, and finally output layer with a Softmax activation function and five neurons since we have five classes. Knowing that after each convolutional layer, we do max pooling for the output. Table 4.4 shows a hyperparameter valuse for the training of the CNN model.

Table 4.4. Hyperparameter values for training of CNN.

| Hyperparameters | Values | | | | |
|---|---|---|---|---|---|
| **Number of epochs** | BERT | ELMo | Word2Vec | GloVe | FastText |
| | 23 | 18 | 20 | 16 | 14 |
| **Batch size** | 256 | | | | |
| **Loss function** | Categorical Cross-entropy | | | | |
| **Activation functions** | Softmax | | | | |

| | |
|---|---|
| **Activation function of CNNs** | RelU |
| **$CNN_1$ filter size** | 50 |
| **$CNN_2$ filter size** | 100 |
| **$CNN_3$ filter size** | 200 |
| **Kernel size of CNNs** | 3 |
| **Dropout of CNNs** | 0.2 |
| **Hidden layer no** | 2 |
| **Activation functions for hidden layer** | RelU |
| **Train Approach** | Cross-validation |
| **Optimizer** | Adam |

#### 4.1.3.3. LSTM

At the beginner is embedding, then flatten layer, LSTM layer with 100 cells, two hidden layers with a RelU activation function, each containing 100 neurons, and finally, output layer with softmax activation function and five neurons. Table 4.5 shows the hyperparameter values for the training of LSTM.

Table 04.5. Hyperparameter values for training of LSTM.

| **Hyperparameters** | **Values** | | | | |
|---|---|---|---|---|---|
| **Number of epochs** | BERT | ELMo | Word2Vec | GloVe | FastText |
| | 12 | 9 | 10 | 9 | 8 |
| **Batch size** | 256 | | | | |
| **Loss function** | Categorical Cross-entropy | | | | |
| **Activation functions** | Softmax | | | | |
| **LSTM cells no** | 100 | | | | |
| **Hidden layer no** | 2 | | | | |
| **Activation functions for hidden layer** | RelU | | | | |
| **Train Approach** | Cross-validation | | | | |

| | |
|---|---|
| **Optimizer** | Adam |

#### 4.1.3.4. BiLSTM

At the beginner is embedding, then flatten layer, BiLSTM layer with 100 cells, and finally, output layer with softmax activation function and five neurons. Table 4.6 shows the hyperparameter values for training of BiLSTM.

Table 04.6. Hyperparameter values for training of BiLSTM.

| Hyperparameters | Values | | | | |
|---|---|---|---|---|---|
| **Number of epochs** | BERT | ELMo | Word2Vec | GloVe | FastText |
| | 11 | 10 | 10 | 9 | 11 |
| **Batch size** | 256 | | | | |
| **Loss function** | Categorical Cross-entropy | | | | |
| **Activation functions** | Softmax | | | | |
| **BiLSTM cells no** | 100 | | | | |
| **Hidden layer no** | 2 | | | | |
| **Activation functions for hidden layer** | RelU | | | | |
| **Train Approach** | Cross-validation | | | | |
| **Optimizer** | Adam | | | | |

### 4.2. SOFTWARE LIBRARIES

This section will be about the most important software libraries used with the classes and functions that are called from those libraries. Table 4.7 shows the software libraries with the classes and functions.

Table 4.7. Software libraries with the classes and functions.

| Library | Definition | Functions & classes |
|---|---|---|
| NLTK | It's a vast library and a shortcut for the Natural | - Stopwords: a function that gives a set of stop words. |

| | | |
|---|---|---|
| | Language Toolkit. It deals with textual Data, is probably used for pre-processing, and contains many functions. | - WordNetLemetizer: This is a function for word lemmatizing. |
| Keras | It is a library and open source written in Python. It is a very large library that can be used for several machines and deep learning tasks. | - Tokenizer: It is to tokenize the data so that each word will have a unique number, and each record becomes a vector of numbers.<br>- Pad_sequences: The vectors will vary depending on the number of words, so we padded to 100 processes to make them all the same length.<br>- Sequential: To create a deep learning model, we call the evaluate method from it to calculate the value of the accuracy and loss of the trained model through which we add the network layers.<br>- Embedding: to create embedding layer.<br>- Conv1D: to create convolution layer.<br>- MaxPooling1D: to create pooling layer.<br>- Dropout: to create dropout layer.<br>- Flatten: to create flatten layer.<br>- Activation: to certain the activation function as a parameter.<br>- Lstm: to create lstm layer.<br>- Bilstm: to create bilstm layer.<br>- Dense: to create normal layer. |

| | | |
|---|---|---|
| Sklearn (Scikit learn) | It is a library written in Python that supports supervised and unsupervised deep learning tasks. | - Kfold: to create the folds of data for training and testing.<br>- Accuracy score: to calculate the value of accuracy.<br>- Confusion matrix: to create the confusion metrix.<br>- Roc_curve: to plot the roc curves for each class.<br>- Precision_score: to calculate the value of precition.<br>- Recall_score: to calculate the value of recall.<br>- F1_score: to calculate the value of F1. |
| Numpy | It is a Python library with which multidimensional arrays can be created and easily apply many mathematical relationships. | - np: class to call out the array function to create the two Numby arrays, one containing the training data and the other containing the test data. |
| Matplotlib | A famous drawing library for drawing diagrams that express information. | - Plt : class to draw the learning craves |
| Genism | It is a Python library created to represent the semantics of texts. | - Word2Vec: to create semantic vectors through Word2Vec model.<br>- FastText: to create semantic vectors through the FastText model. |

**PART 5**

**SUMMARY**

## 5.1. RESULTS & DISCUSSION

In this section, the results will be presented and discussed, where the values of evaluation metrics as ACC, error rate, recall, precision, F1-score, AUC, and MCC for CNN, LSTM, BILST, and DFFNN as DL model, and also for Word2Vec, FastText, GloVe, ELMo, and BERT as language model are shown in Table 5.1.

Table 05.1. Evaluation metrics for all DL models and language models used.

| Language Model | Classifier | Acc | Error Rate | Recall | Precision | F1-Score | AUC | MCC |
|---|---|---|---|---|---|---|---|---|
| Woed2Vec | CNN | 0.91 | 0.09 | 0.86 | 0.86 | 0.86 | 0.91 | 0.80 |
| | DFFNN | 0.89 | 0.11 | 0.83 | 0.83 | 0.83 | 0.89 | 0.75 |
| | LSTM | 0.91 | 0.09 | 0.86 | 0.86 | 0.86 | 0.91 | 0.80 |
| | BiLSTM | 0.91 | 0.09 | 0.86 | 0.86 | 0.86 | 0.91 | 0.79 |
| Means: | | 0.91 | 0.10 | 0.85 | 0.85 | 0.85 | 0.91 | 0.79 |
| FastText | CNN | 0.91 | 0.09 | 0.86 | 0.86 | 0.86 | 0.91 | 0.79 |
| | DFFNN | 0.90 | 0.10 | 0.86 | 0.86 | 0.86 | 0.89 | 0.78 |
| | LSTM | 0.90 | 0.10 | 0.86 | 0.86 | 0.86 | 0.91 | 0.79 |
| | BiLSTM | 0.91 | 0.09 | 0.86 | 0.86 | 0.86 | 0.91 | 0.80 |
| Means: | | 0.91 | 0.10 | 0.86 | 0.86 | 0.86 | 0.91 | 0.79 |
| GloVe | CNN | 0.92 | 0.08 | 0.88 | 0.88 | 0.88 | 0.93 | 0.81 |
| | DFFNN | 0.91 | 0.09 | 0.87 | 0.87 | 0.87 | 0.89 | 0.80 |
| | LSTM | 0.91 | 0.09 | 0.87 | 0.87 | 0.87 | 0.92 | 0.80 |
| | BiLSTM | 0.91 | 0.09 | 0.87 | 0.87 | 0.87 | 0.92 | 0.80 |
| Means: | | 0.91 | 0.09 | 0.87 | 0.87 | 0.87 | 0.92 | 0.80 |
| ELMo | CNN | 0.93 | 0.07 | 0.90 | 0.90 | 0.90 | 0.94 | 0.85 |
| | DFFNN | 0.93 | 0.07 | 0.89 | 0.89 | 0.89 | 0.93 | 0.83 |
| | LSTM | 0.92 | 0.08 | 0.88 | 0.88 | 0.88 | 0.93 | 0.82 |
| | BiLSTM | 0.92 | 0.08 | 0.88 | 0.88 | 0.88 | 0.93 | 0.82 |
| Means: | | 0.93 | 0.08 | 0.89 | 0.89 | 0.89 | 0.93 | 0.83 |
| BERT | CNN | 0.95 | 0.05 | 0.92 | 0.92 | 0.92 | 0.96 | 0.88 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | DFFNN | 0.95 | 0.05 | 0.92 | 0.92 | 0.92 | 0.96 | 0.89 |
| | LSTM | 0.94 | 0.06 | 0.91 | 0.91 | 0.91 | 0.95 | 0.87 |
| | BiLSTM | 0.94 | 0.06 | 0.91 | 0.91 | 0.91 | 095 | 0.87 |
| Means: | | 0.95 | 0.06 | 0.92 | 0.92 | 0.92 | 0.96 | 0.89 |

It was noted that the highest performance was obtained when BERT was used as a language model, specifically with DFFNN and CNN as a DL model, where a 95% was obtained as a test accuracy and 92% as a recall value. As for the training models compared between them, it was noted that they are approximately similar in performance, but most of the time, DFFNN and CNN were the highest.

In the beginning, feelings were classified into five categories. Still, a high performance wasn't obtained due to the difficulty of distinguishing between the reviews that were satisfied and very satisfied and the reviews that were not satisfied and not very satisfied because of the closeness and similarity of the significant meaning between them; it is also due to the lack of people rated with three, four or five stars, which led to unbalanced data. That is why in this study, classification has been made into three types of emotions.

Table 5.2 shows the accuracy value for each class and the $BERT_{base}$ model with DFFNN. We note the low recall value of classification for the Neutral class and the high recall value for classes Unsatisfied and Satisfied.

Table 5.2. The performance values for each class using the $BERT_{base}$ and DFFNN.

| | **Unsatisfied** | **Neutral** | **Satisfied** |
|---|---|---|---|
| Accuracy | 0.95 | 0.95 | 0.94 |
| Recall | 0.90 | 0.55 | 0.97 |
| Precision | 0.89 | 0.79 | 0.94 |
| F1-score | 0.90 | 0.65 | 0.96 |

It is possible to improve the performance accuracy by merging the two and one-rated items by making them one class, merging the four and five-rated items as one class, and deleting items with a 3-star rating. Thus our issue turns into a classification of two items (binary classification) and obtaining a higher accuracy.

Through experimentation, it was noted number five is the best value for choosing the number of folders in the cross-validation, where in this case, the test section is 0.2, and the training section is 0.8 from the complete dataset in each iteration. , It was also noted that the model's performance is approximately the same in all iterations. Figure 5.1 shows ROC curves for each class and iteration, where the classification done by FastText and CNN. In our project and the case of our dataset, performance didn't make a difference if we used cross-validation; the performance was the same.

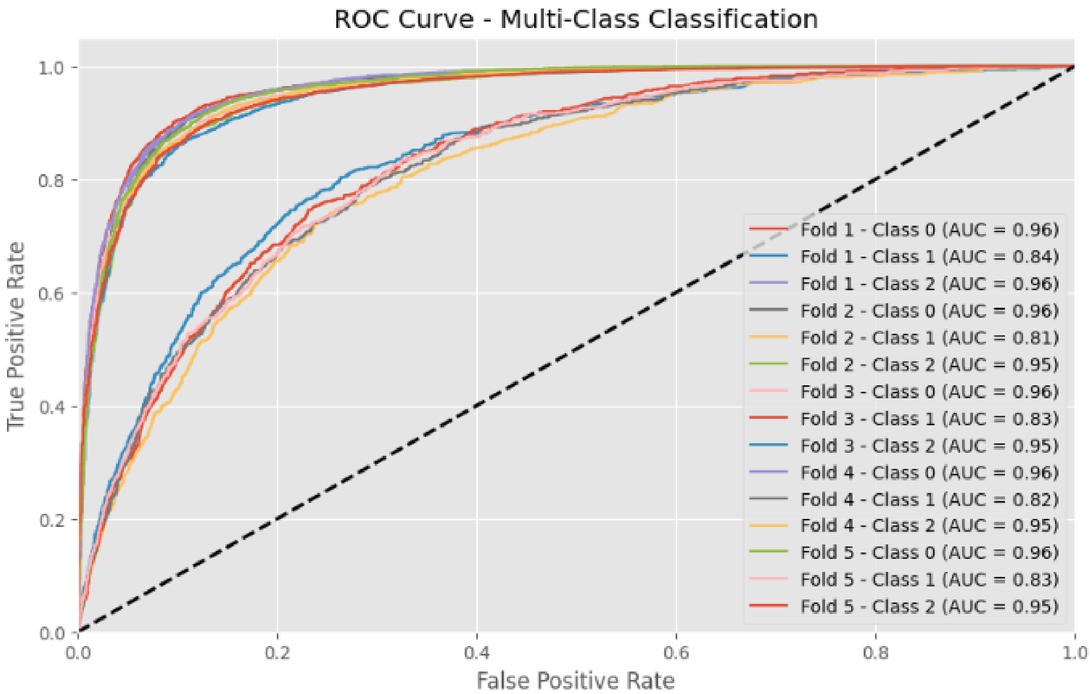


Figure 5.1. ROC curves for each class and iteration using FastText and CNN.

The selection of hyperparameters in this study was chosen experimentally during the training process. Also, the great role of hyperparameters in the classification and training process was concluded from this study and how the selection of these coefficients must be accurate and careful to select them, for example when the number of training epochs is small, little accuracy was got, where the classification model was CNN, so selecting these parameters as carefully is an essential thing in the classification process.

It was noted that the great role of CLMs in performance in text analysis projects in terms of word representation and DL, the creation of semantic vectors, and then they become features of the DL model and how they far outperformed SLMs.

## 5.2. CONCLUSION

This research suggests high-performance SA methods that combine DL and word representation models that have already been trained. In these methods, the input text data is represented by five state-of-the pre-trained models, including BERTbase and ELMo as CLM, Word2Vec, FastText, and GloVe as SLM.

As a result, we demonstrated that SA models perform better when DL architectures are used with language models to effectively capture the semantic meaning and context of words in the text. The study showed that current language models, including BERTbase, and ELMo, performed better and more accurately in this task than traditional word embeddings as SLMs.

The study also emphasized hyperparameters' importance in figuring out the models' accuracy. Overall, these findings highlight the challenges of SA in multi classes and the advantages of employing CLMs that consider the precise context in which a term is employed.

In addition, this research has demonstrated that when contextualizing input texts and vectorizing them by a semantic and distribution space, the performance of language models with various architectures and training sets greatly influences the classification operations. That features, which are semantic vectors extracted from the text, how they had a significant and valuable role in classification, and how DL took them into account. Also, in this study, the effectiveness of DL models as a classifier in NLP applications was shown, and how each DL can partner with the NLP to solve realistic and multiple problems.

Among the future work that we can do to develop our proposed study, it is possible to combine two language models to represent words and create semantic vectors, it is possible to rely on pre-trained models in the classification process itself, such as the BERT classification model, and it is also possible to implement different techniques in feature engineering and apply them to our features.

## REFERENCES


1. N. K. Chauhan and K. Singh, “A review on conventional machine learning vs deep learning,” in *2018 International conference on computing, power and communication technologies (GUCON)*, 2018, pp. 347–352.

2. A. Mohamed, J. Ren, M. El-Gindy, H. Lang, and A. N. Ouda, “Literature survey for autonomous vehicles: sensor fusion, computer vision, system identification and fault tolerance,” *Int. J. Autom. Control*, vol. 12, no. 4, pp. 555–581, 2018.

3. X. Dai, “HybridNet: A fast vehicle detection system for autonomous driving,” *Signal Process. Image Commun.*, vol. 70, pp. 79–88, 2019.

4. E. M. Mercha and H. Benbrahim, “Machine learning and deep learning for sentiment analysis across languages: A survey,” *Neurocomputing*, vol. 531, pp. 195–216, 2023.

5. Q. Qiu, M. Tian, K. Ma, Y. J. Tan, L. Tao, and Z. Xie, “A question answering system based on mineral exploration ontology generation: A deep learning methodology,” *Ore Geol. Rev.*, p. 105294, 2023.

6. A. Al-Sadi, M. Al-Ayyoub, Y. Jararweh, and F. Costen, “Visual question answering in the medical domain based on deep learning approaches: A comprehensive study,” *Pattern Recognit. Lett.*, vol. 150, pp. 57–75, 2021.

7. M. Bhattacharya, S. Bhat, S. Tripathy, A. Bansal, and M. Choudhary, “Improving biomedical named entity recognition through transfer learning and asymmetric tri-training,” *Procedia Comput. Sci.*, vol. 218, pp. 2723–2733, 2023.

8. A. Goyal, V. Gupta, and M. Kumar, “A deep learning-based bilingual Hindi and Punjabi named entity recognition system using enhanced word embeddings,” *Knowledge-Based Syst.*, vol. 234, p. 107601, 2021.

9. P. Mehta and S. Pandya, “A review on sentiment analysis methodologies, practices and applications,” *Int. J. Sci. Technol. Res.*, vol. 9, no. 2, pp. 601–609, 2020.

10. M. E. Basiri and A. Kabiri, “Sentence-level sentiment analysis in Persian,” in *2017 3rd International Conference on Pattern Recognition and Image Analysis (IPRIA)*, 2017, pp. 84–89.

11. A. Sharma and S. Dey, “A document-level sentiment analysis approach using artificial neural network and sentiment lexicons,” *ACM SIGAPP Appl. Comput. Rev.*, vol. 12, no. 4, pp. 67–75, 2012.

12. N. Nandal, R. Tanwar, and J. Pruthi, “Machine learning based aspect level sentiment analysis for Amazon products,” *Spat. Inf. Res.*, vol. 28, pp. 601–607, 2020.

13. D. Antypas, A. Preece, and J. Camacho-Collados, “Negativity spreads faster: A large-scale multilingual twitter analysis on the role of sentiment in political communication,” *Online Soc. Networks Media*, vol. 33, p. 100242, 2023.

14. M. M. Hasan and H. Jiang, “Political sentiment and corporate social responsibility,” *Br. Account. Rev.*, vol. 55, no. 1, p. 101170, 2023.

15. A. R. Rahmanti *et al.*, “Social media sentiment analysis to monitor the performance of vaccination coverage during the early phase of the national COVID-19 vaccine rollout,” *Comput. Methods Programs Biomed.*, vol. 221, p. 106838, 2022.

16. R. Haque, N. Islam, M. Tasneem, and A. K. Das, “MULTI-CLASS SENTIMENT CLASSIFICATION ON BENGALI SOCIAL MEDIA COMMENTS USING MACHINE LEARNING,” *Int. J. Cogn. Comput. Eng.*, 2023.

17. C. Qian, N. Mathur, N. H. Zakaria, R. Arora, V. Gupta, and M. Ali, “Understanding public opinions on social media for financial sentiment analysis using AI-based techniques,” *Inf. Process. Manag.*, vol. 59, no. 6, p. 103098, 2022.

18. H.-C. K. Lin, T.-H. Wang, G.-C. Lin, S.-C. Cheng, H.-R. Chen, and Y.-M. Huang, “Applying sentiment analysis to automatically classify consumer comments concerning marketing 4Cs aspects,” *Appl. Soft Comput.*, vol. 97, p. 106755, 2020.

19. D. Sunitha, R. K. Patra, N. V Babu, A. Suresh, and S. C. Gupta, “Twitter sentiment analysis using ensemble based deep learning model towards COVID-19 in India and European countries,” *Pattern Recognit. Lett.*, vol. 158, pp. 164–170, 2022.

20. N. Leelawat *et al.*, “Twitter data sentiment analysis of tourism in Thailand during the COVID-19 pandemic using machine learning,” *Heliyon*, vol. 8, no. 10, p. e10894, 2022.

21. L. Abualigah, H. E. Alfar, M. Shehab, and A. M. A. Hussein, “Sentiment analysis in healthcare: a brief review,” *Recent Adv. NLP case Arab. Lang.*, pp. 129–141, 2020.

22. K. Korovkinas, P. Danenas, and G. Garšva, “SVM and k-Means Hybrid Method for Textual Data Sentiment Analysis.,” *Balt. J. Mod. Comput.*, vol. 7, no. 1, 2019.

23. A. S. M. AlQahtani, “Product sentiment analysis for amazon reviews,” *Int. J. Comput. Sci. Inf. Technol. Vol*, vol. 13, 2021.

24. S. A. Aljuhani and N. S. Alghamdi, "A comparison of sentiment analysis methods on Amazon reviews of Mobile Phones," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 6, 2019.

25. J. Sangeetha and U. Kumaran, "Sentiment analysis of amazon user reviews using a hybrid approach," *Meas. Sensors*, vol. 27, p. 100790, 2023.

26. B. Bansal and S. Srivastava, "Sentiment classification of online consumer reviews using word vector representations," *Procedia Comput. Sci.*, vol. 132, pp. 1147–1153, 2018.

27. L. Zhang, K. Hua, H. Wang, G. Qian, and L. Zhang, "Sentiment analysis on reviews of mobile users," *Procedia Comput. Sci.*, vol. 34, pp. 458–465, 2014.

28. M. A. Fauzi, "Random forest approach fo sentiment analysis in indonesian," *Indones. J. Electr. Eng. Comput. Sci*, vol. 12, pp. 46–50, 2018.

29. A. Borg and M. Boldt, "Using VADER sentiment and SVM for predicting customer response sentiment," *Expert Syst. Appl.*, vol. 162, p. 113746, 2020.

30. T. H. J. Hidayat, Y. Ruldeviyani, A. R. Aditama, G. R. Madya, A. W. Nugraha, and M. W. Adisaputra, "Sentiment analysis of twitter data related to Rinca Island development using Doc2Vec and SVM and logistic regression as classifier," *Procedia Comput. Sci.*, vol. 197, pp. 660–667, 2022.

31. M. Bibi *et al.*, "A novel unsupervised ensemble framework using concept-based linguistic methods and machine learning for twitter sentiment analysis," *Pattern Recognit. Lett.*, vol. 158, pp. 80–86, 2022.

32. Y. Zhang, J. Wang, and X. Zhang, "Conciseness is better: recurrent attention lstm model for document-level sentiment analysis," *Neurocomputing*, vol. 462, pp. 101–112, 2021.

33. D. O. Oyewola, L. A. Oladimeji, S. O. Julius, L. B. Kachalla, and E. G. Dada, "Optimizing sentiment analysis of Nigerian 2023 presidential election using two-stage residual long short term memory," *Heliyon*, p. e14836, 2023.

34. W. Li, L. Zhu, Y. Shi, K. Guo, and E. Cambria, "User reviews: Sentiment analysis using lexicon integrated two-channel CNN–LSTM family models," *Appl. Soft Comput.*, vol. 94, p. 106435, 2020.

35. D. Maity, S. Kanakaraddi, and S. Giraddi, "Text Sentiment Analysis based on Multichannel Convolutional Neural Networks and Syntactic Structure," *Procedia Comput. Sci.*, vol. 218, pp. 220–226, 2023.

36. M. P. Geetha and D. K. Renuka, "Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model," *Int. J. Intell. Networks*, vol. 2, pp. 64–69, 2021.

37. A. Patel, P. Oza, and S. Agrawal, "Sentiment Analysis of Customer Feedback and Reviews for Airline Services using Language Representation Model," *Procedia Comput. Sci.*, vol. 218, pp. 2459–2467, 2023.

38. I. N. Khasanah, "Sentiment classification using fasttext embedding and deep learning model," *Procedia Comput. Sci.*, vol. 189, pp. 343–350, 2021.

39. P. F. Muhammad, R. Kusumaningrum, and A. Wibowo, "Sentiment analysis using Word2vec and long short-term memory (LSTM) for Indonesian hotel reviews," *Procedia Comput. Sci.*, vol. 179, pp. 728–735, 2021.

40. M. Siddharth and R. Aarthi, "Blended multi-class text to image synthesis GANs with RoBerTa and Mask R-CNN," *Procedia Comput. Sci.*, vol. 218, pp. 845–857, 2023.

41. K. Kaur and P. Kaur, "BERT-CNN: Improving BERT for Requirements Classification using CNN," *Procedia Comput. Sci.*, vol. 218, pp. 2604–2611, 2023.

42. N. Badri, F. Kboubi, and A. H. Chaibi, "Combining FastText and Glove word embedding for offensive and hate speech text detection," *Procedia Comput. Sci.*, vol. 207, pp. 769–778, 2022.

43. K. Chowdhary and K. R. Chowdhary, "Natural language processing," *Fundam. Artif. Intell.*, pp. 603–649, 2020.

44. K. M. KARAOĞLAN, "ÖZELLİK TABANLI GÖRÜŞ MADENCİLİĞİNDE YAPAY ZEKA TEKNİKLERİ KULLANARAK GÖRÜŞ HEDEFİ ÇIKARIMI VE KATEGORİ TESPİTİ." 2022.

45. A. Balahur and M. Turchi, "Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis," *Comput. Speech Lang.*, vol. 28, no. 1, pp. 56–75, 2014.

46. N. T. Rudrappa, M. V Reddy, and M. Hanumanthappa, "HiTEK Pre-processing for Speech and Text: NLP," *Indian J. Sci. Technol.*, vol. 16, no. 19, pp. 1413–1421, 2023.

47. D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: State of the art, current trends and challenges," *Multimed. Tools Appl.*, vol. 82, no. 3, pp. 3713–3744, 2023.

48. M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artif. Intell. Rev.*, vol. 55, no. 7, pp. 5731–5780, 2022.

49. K. Ravi and V. Ravi, "A survey on opinion mining and sentiment analysis: tasks, approaches and applications," *Knowledge-based Syst.*, vol. 89, pp. 14–46, 2015.

50. P. Ray and A. Chakrabarti, “A mixed approach of deep learning method and rule-based method to improve aspect level sentiment analysis,” *Appl. Comput. Informatics*, vol. 18, no. 1/2, pp. 163–178, 2022.

51. L. M. Gómez and M. N. Cáceres, “Applying data mining for sentiment analysis in music,” in *Trends in Cyber-Physical Multi-Agent Systems. The PAAMS Collection-15th International Conference, PAAMS 2017 15*, 2018, pp. 198–205.

52. S. Zad, M. Heidari, J. H. Jones, and O. Uzuner, “A survey on concept-level sentiment analysis techniques of textual data,” in *2021 IEEE World AI IoT Congress (AIIoT)*, 2021, pp. 285–291.

53. J. Bagherzadeh and H. Asil, “A review of various semi-supervised learning models with a deep learning and memory approach,” *Iran J. Comput. Sci.*, vol. 2, pp. 65–80, 2019.

54. B. Mahesh, “Machine learning algorithms-a review,” *Int. J. Sci. Res. (IJSR).[Internet]*, vol. 9, no. 1, pp. 381–386, 2020.

55. K. K. Mohbey, “Sentiment analysis for product rating using a deep learning approach,” in *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, 2021, pp. 121–126.

56. Y. Indulkar and A. Patil, “Sentiment Analysis of Uber & Ola using Deep Learning,” in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 2020, pp. 21–27.

57. K. Sundus, F. Al-Haj, and B. Hammo, “A deep learning approach for arabic text classification,” in *2019 2nd International Conference on New Trends in Computing Sciences (ICTCS)*, 2019, pp. 1–7.

58. A. Vassilev, “Bowtie-a deep learning feedforward neural network for sentiment analysis,” in *Machine Learning, Optimization, and Data Science: 5th International Conference, LOD 2019, Siena, Italy, September 10–13, 2019, Proceedings 5*, 2019, pp. 360–371.

59. M. Al-Smadi, B. Talafha, M. Al-Ayyoub, and Y. Jararweh, “Using long short-term memory deep neural networks for aspect-based sentiment analysis of Arabic reviews,” *Int. J. Mach. Learn. Cybern.*, vol. 10, pp. 2163–2175, 2019.

60. Y. Yu, X. Si, C. Hu, and J. Zhang, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, 2019.

61. K. Zhang, W. Song, L. Liu, X. Zhao, and C. Du, “Bidirectional long short-term memory for sentiment analysis of Chinese product reviews,” in *2019 IEEE 9th international conference on electronics information and emergency communication (ICEIEC)*, 2019, pp. 1–4.

62. G. Liu and J. Guo, “Bidirectional LSTM with attention mechanism and convolutional layer for text classification,” *Neurocomputing*, vol. 337, pp. 325–338, 2019.

63. A. Severyn and A. Moschitti, “Twitter sentiment analysis with deep convolutional neural networks,” in *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, 2015, pp. 959–962.

64. L. Alzubaidi *et al.*, “Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions,” *J. big Data*, vol. 8, pp. 1–74, 2021.

65. Z. A. Sejuti and M. S. Islam, “A hybrid CNN–KNN approach for identification of COVID-19 with 5-fold cross validation,” *Sensors Int.*, vol. 4, p. 100229, 2023.

66. K. Mohamad and K. M. Karaoğlan, “Enhancing Deep Learning-Based Sentiment Analysis Using Static and Contextual Language Models,” *Bitlis Eren Univ. J. Sci.*, vol. 12, p. 13, 2023.

67. K. M. Karaoğlan and O. Fındık, “Extended rule-based opinion target extraction with a novel text pre-processing method and ensemble learning,” *Appl. Soft Comput.*, vol. 118, p. 108524, 2022.

68. Z. Singla, S. Randhawa, and S. Jain, “Sentiment analysis of customer product reviews using machine learning,” in *2017 international conference on intelligent computing and control (I2C2)*, 2017, pp. 1–5.

69. Y. Sun, H. Qiu, Y. Zheng, Z. Wang, and C. Zhang, “SIFRank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model,” *IEEE Access*, vol. 8, pp. 10896–10906, 2020.

70. S. Al-Saqqa and A. Awajan, “The use of word2vec model in sentiment analysis: A survey,” in *Proceedings of the 2019 international conference on artificial intelligence, robotics and control*, 2019, pp. 39–43.

71. G. Di Gennaro, A. Buonanno, and F. A. N. Palmieri, “Considerations about learning Word2Vec,” *J. Supercomput.*, pp. 1–16, 2021.

72. S. Hu, A. Kumar, F. Al-Turjman, S. Gupta, and S. Seth, “Reviewer credibility and sentiment analysis based user profile modelling for online product recommendation,” *IEEE Access*, vol. 8, pp. 26172–26189, 2020.

73. I. Santos, N. Nedjah, and L. de Macedo Mourelle, “Sentiment analysis using convolutional neural network with fastText embeddings,” in *2017 IEEE Latin American conference on computational intelligence (LA-CCI)*, 2017, pp. 1–5.

74. Y. Sharma, G. Agrawal, P. Jain, and T. Kumar, “Vector representation of words for sentiment analysis using GloVe,” in *2017 international conference on intelligent communication and computational techniques (icct)*, 2017, pp. 279–284.

75. A. Konstantinov, V. Moshkin, and N. Yarushkina, “Approach to the use of language models BERT and Word2vec in sentiment analysis of social network texts,” in *International Scientific and Practical Conference in Control Engineering and Decision Making*, 2020, pp. 462–473.

76. G. Paaß and S. Giesselbach, “Pre-trained Language Models,” in *Foundation Models for Natural Language Processing: Pre-trained Language Models Integrating Media*, Springer, 2023, pp. 19–78.

77. V. Jain, R. K. Kaliyar, A. Goswami, P. Narang, and Y. Sharma, “AENeT: an attention-enabled neural architecture for fake news detection using contextual features,” *Neural Comput. Appl.*, vol. 34, no. 1, pp. 771–782, 2022.

78. M. G. Sousa, K. Sakiyama, L. de Souza Rodrigues, P. H. Moraes, E. R. Fernandes, and E. T. Matsubara, “BERT for stock market sentiment analysis,” in *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)*, 2019, pp. 1597–1601.

79. M. Singh, A. K. Jakhar, and S. Pandey, “Sentiment analysis on the impact of coronavirus in social life using the BERT model,” *Soc. Netw. Anal. Min.*, vol. 11, no. 1, p. 33, 2021.

80. C. Mastronardo and F. Tamburini, “Enhancing a Text Summarization System with ELMo.,” in *CLiC-it*, 2019.

81. S. Edunov, A. Baevski, and M. Auli, “Pre-trained language model representations for language generation,” *arXiv Prepr. arXiv1903.09722*, 2019.

# RESUME

Khadija MOHAMAD graduated first, elementary education and High School, after that, she started undergraduate program in Damascus University Department of Computer Engineering in 2019. Then in 2020, she started M. Sc. education in Karabük University Department of Computer Engineering in Türkiye.