People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University of 8 May 1945-Guelma-

Faculty of Mathematics, Computer Science, and Science of Matter

Department of Computer Science

Master Thesis

Specialty: Computer Science

Option: Science and technology of information and communication

Theme:

Sentiment Analysis of Algerian Dialect Using Machine Learning Techniques

Presented by :

Mohammed Raid Elislam BOUKHEROUBA

Supervised by :

Dr.Djalila BOUGHAREB

June 2024

Acknowledgments

I would like to begin by expressing my deepest gratitude to Allah for granting me the strength, resilience, and wisdom to undertake this challenging endeavor.

I would like to express my deepest gratitude to my supervisor, Dr. Djalila Boughareb, for their invaluable guidance, unwavering support, and insightful feedback throughout the entire journey of this thesis. Their expertise and encouragement have been instrumental in shaping the direction of my research and enhancing its quality.

I am also thankful to the members of my thesis committee for their valuable input, constructive criticism, and encouragement. Their diverse perspectives have enriched my work and contributed significantly to its development.

I extend my appreciation to my university for providing the resources and facilities necessary for conducting this research. Additionally, I am grateful to the staff and faculty members who have assisted me during my academic pursuits.

I am indebted to my family for their unconditional love, support, and understanding throughout this endeavor. Their encouragement has been my source of strength during challenging times.

Finally, I would like to thank my friends for their encouragement, understanding, and motivation. Their unwavering belief in me has been a driving force behind completing this thesis.

This work would not have been possible without the support and encouragement of all those mentioned above. Thank you for being a part of this journey.

ABSTRACT

Amidst the current global landscape, sentiment analysis has gained remarkable significance across diverse domains spanning social, economic, sports, political, and commercial sectors. The profusion of varied textual content on social media platforms underscores the importance of comprehending these narratives, crucial for deciphering societal trends, meeting citizen needs, and mitigating offensive content. This significance amplifies particularly in dialectal languages characterized by nuanced variations in morphology and meanings.

This master thesis aims to address the challenges posed by the limited size of dialectal Algerian datasets and enhance the accuracy of sentiment detection. The primary objective of this research is to compare the performance of different machine learning, deep learning, and transformers classifiers in analyzing and categorizing a collection of social media posts in the Algerian dialect into two classes: positive or negative sentiment.

By leveraging a novel dataset and employing advanced methodologies, the study aims to provide insights into the efficacy of various algorithms in this task. This endeavor contributes towards advancing sentiment analysis research within the Algerian dialect context, offering valuable implications for real-world applications and decision-making processes.

The achieved results are highly promising, showcasing the best accuracy rate of 87.9% in the Bert base model. In addition to an 85% accuracy employing the LSTM model and 82% accuracy with the SVM linear classifier.

Moreover, the ensemble approaches, which employed aragpt2, arabicBert base, and distilbert, further improved the performance. The majority voting ensemble achieved an accuracy of 90%, and the stacking ensemble attained an accuracy of 91.1%.

Keywords: Opinion Mining, Sentiment Analysis, Deep Learning, Machine Learning, Ensemble Learning, Transformer, Algerian Dialect, Annotated Corpus.

RESUME

Dans le paysage mondial actuel, l'analyse des sentiments a acquis une importance remarquable dans divers domaines tels que les secteurs social, économique, sportif, politique et commercial. La profusion de contenus textuels variés sur les plateformes de médias sociaux souligne l'importance de comprendre ces récits, cruciale pour déchiffrer les tendances sociétales, répondre aux besoins des citoyens et atténuer les contenus offensants. Cette importance se renforce particulièrement dans les langues dialectales caractérisées par des variations nuancées de morphologie et de significations.

Cette thèse de master vise à relever les défis posés par la taille limitée des ensembles de données dialectales algériennes et à améliorer la précision de la détection des sentiments. L'objectif principal de cette recherche est de comparer la performance de différents classificateurs d'apprentissage automatique, d'apprentissage profond et de transformateurs dans l'analyse et la catégorisation d'une collection de publications sur les réseaux sociaux en dialecte algérien en deux classes : sentiment positif ou négatif.

En exploitant un nouvel ensemble de données et en utilisant des méthodologies avancées, l'étude vise à fournir des informations sur l'efficacité de divers algorithmes dans cette tâche. Cette démarche contribue à faire progresser la recherche sur l'analyse des sentiments dans le contexte du dialecte algérien, offrant des implications précieuses pour les applications réelles et les processus de prise de décision.

Les résultats obtenus sont très prometteurs, montrant le meilleur taux de précision de 87,9 % avec le modèle Bert de base, en plus d'une précision de 85 % avec le modèle LSTM et de 82 % avec le classificateur linéaire SVM. De plus, les approches ensemblistes, qui ont utilisé aragpt2, arabicBert de base et distilbert, ont encore amélioré les performances. L'ensemble de vote majoritaire a atteint une précision de 90 %, et l'ensemble d'empilement a atteint une précision de 91,1 %.

Mots-clés : Fouille d'Opinions, Analyse des Sentiments, Apprentissage Profond, Apprentissage Automatique, Apprentissage par Ensemble, Transformateur, Dialecte Algérien, Corpus Annoté.

الملخص

في ظل المشهد العالمي الحالي، اكتسب تحليل المشاعر أهمية كبيرة عبر مجالات متنوعة تشمل القطاعات الاجتماعية والاقتصادية والرياضية والسياسية والتجارية. تؤكد وفرة المحتوى النصي المتنوع على منصات التواصل الاجتماعي على أهمية فهم هذه السرديات، الأمر الذي يعد ضرورياً لفهم الاتجاهات المجتمعية وتلبية احتياجات المواطنين والحد من المحتوى المسيء. تتزايد هذه الأهمية بشكل خاص في اللغات العامية التي تتميز بتنوع دقيق في الصرف والمعاني.

تهدف هذه الأطروحة إلى معالجة التحديات التي يفرضها الحجم المحدود لمجموعات البيانات العامية الجزائرية وتعزيز دقة اكتشاف المشاعر. الهدف الرئيسي من هذا البحث هو مقارنة أداء مختلف المصنفات في التعلم الآلي والتعلم العميق والمحولات في تحليل وتصنيف مجموعة من منشورات وسائل التواصل الاجتماعي باللهجة الجزائرية إلى فئتين: المشاعر الإيجابية أو السلبية. من خلال الاستفادة من مجموعة بيانات جديدة وتطبيق منهجيات متقدمة، تهدف الدراسة إلى تقديم رؤى حول فعالية الخوارزميات المختلفة في هذه المهمة. تساهم هذه الجهود في دفع عجلة البحث في تحليل المشاعر ضمن سياق اللهجة الجزائرية، مما يوفر دلالات قيمة للتطبيقات الواقعية وعمليات اتخاذ القرار.

النتائج التي تم تحقيقها واعدة للغاية، حيث حققت نموذج بيرت الأساسي أفضل معدل دقة بنسبة 87.9%. بالإضافة إلى دقة 85% باستخدام نموذج LSTM ودقة 82% مع المصنف الخطي SVM.

علاوة على ذلك، فإن النهج التجميعي، الذي استخدم aragpt2 وarabicBert base وdistilbert، حسن الأداء بشكل أكبر. حقق تجميع التصويت بالأغلبية دقة بلغت 90%، وحقق تجميع التكديس دقة بلغت 91.1%.

الكلمات المفتاحية: استخراج الآراء، تحليل المشاعر، التعلم العميق، التعلم الآلي، التعلم التجميعي، المحول، اللهجة الجزائرية، مجموعة البيانات المشروحة.

List of abbreviations

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

General introduction

In today's world, characterized by a vast volume of unstructured textual data from digital communications and social media interactions, sentiment analysis has gained significant importance across multiple domains, revolutionizing how organizations and industries interpret and utilize customer opinions, social media trends, and market dynamics. By extracting meaningful insights from this data, sentiment analysis helps businesses enhance customer satisfaction, manage brand reputation, and make informed strategic decisions. Additionally, it aids in market predictions and risk management in finance, gauges public opinion in politics, and improves patient care in healthcare.

However, Sentiment analysis of the Algerian dialect presents several challenges due to the nature of social media interactions and the specific characteristics of the dialect. Social media comments often contain unstructured language, lacking capital letters typically used to identify features. These texts are rife with spelling errors, slang, abbreviations, and repeated letters to emphasize words, complicating the preprocessing stage. Additionally, there is a significant scarcity of open-source datasets for the Algerian dialect, forcing researchers to collect and annotate their own data.

The primary objective of this thesis is to improve the accuracy of detecting sentiment in Algerian dialectal comments. To achieve this, we propose to compare the performance of various classifiers in binary sentiment analysis. This comparison includes three main categories of classifiers: transformer-based models (e.g., BERT and GPT), traditional machine learning models (e.g., Support Vector Machines (SVM), k-Nearest Neighbors (KNN), and Naive Bayes), and deep learning models (e.g., Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN)). Furthermore, we delve into ensemble learning techniques, employing both majority voting and stacking methodologies, to further augment the performance of our transformer models.

The motivation behind this research is to determine which models, or combinations of models, deliver the best performance for sentiment analysis tasks, especially when dealing with diverse and complex datasets.

Master's thesis structure:

This master's thesis is structured into three chapters as follows:

- The first chapter delves into the field of sentiment analysis, highlighting the significance of dialect and examining the challenges of the Algerian dialect, such as data scarcity and complex writing systems. It concludes by exploring effective methods to overcome these challenges
- The second chapter outlines our approach to comparing sentiment analysis models on an Algerian Arabic dialect corpus, covering the entire process from data collection and preprocessing to model construction, parameter tuning, and final results.
- The third chapter delves into the implementation of the transformers, machine learning, and deep learning classifiers utilized to develop our sentiment analysis models, featuring popular frameworks and code examples.

CHAPTER 1: Challenges and Solutions in Sentiment Analysis of Algerian Dialect

1.1 Introduction

In this chapter, we will focus on defining sentiment analysis and its most important variations, as well as shed light on dialects, highlighting their significance and diverse types. Sequentially, we will address the unique characteristics of the Algerian dialect and the most important challenges it faces, whether the problems relate to the availability of data, the difficulty of classifying this data due to the complexities of the language and its diversity at the linguistic level, or through the diversity of its writing, in Latin or Arabic letters. Furthermore, we'll dissect the most prominent factors that affect the interpretation of these feelings, spanning from linguistic nuances to social contexts. Finally, we will explore the most important methods and techniques used to address the difficulties mentioned previously, ensuring a comprehensive understanding of sentiment analysis in diverse linguistic landscapes.

1.2 Sentiment analysis

Sentiment analysis is a subset of opinion mining; it is a technique in natural language processing research used to find and extract subjective information from text documents (books, films, products, etc.). It generally seeks to ascertain the writer's feelings toward a particular element or the document's overall contextual polarity, and then classify it into separate categories (generally positive or negative) using different algorithms [1, 2].

1.2.1 Sentiment Analysis Types

There are many types of sentiment analysis, and we will mention the most prominent ones [3]:

- **Fine-grained sentiment**: This is a straightforward and widely used method for grasping customer opinions. It involves examining feedback to discern sentiments, typically categorized as positive, neutral, or negative. Another approach involves rating feedback on a scale of 1 to 5. Many e-commerce platforms employ these techniques to gauge customer sentiment.
- **Emotion detection and sentiment analysis:** emotional sentiment analysis determines the prevailing emotions in a text (e.g., sadness or joy) to gauge overall sentiment. It categorizes text based on emotional content to infer the underlying mood or tone.
- **Aspect-based analysis:** this type of sentiment analysis is concentrated on examining specific aspects of a given product or service. Aspect-based sentiment analysis is crucial for organizations as it aids in automatically organizing and analyzing customer data, streamlining processes like customer support tasks, and providing valuable real-time

insights. It enables organizations to focus on the aspects of their products or services that customers are complaining about, facilitating the progressive resolution of issues. This includes addressing complaints such as technical glitches or major bugs in newly developed software applications.

- **Intent-based sentiment analysis:** intent categorization involves automatically sorting textual data according to the intended purpose of the customer. An intent classifier can analyze written communications and categorize them into intents such as purchase, downgrade, unsubscribe, and others. This capability is valuable for understanding the underlying motives behind a multitude of customer inquiries, streamlining processes, and gaining meaningful insights.

1.3 Dialect

In the simplest terms possible, a dialect is a non-standard use of a particular language or linguistic variation, spoken by a group of speakers, that is indicated by systematic markers such as syntactical, phonological, and grammatical markers [4]. The Oxford English Dictionary defines dialect as ‘Manner of speaking, language, speech; esp. A manner of speech peculiar to, or characteristic of, a particular person or class’ (1989).

In essence, a dialect refers to the subcategorization of a language that deviates from the standard form and is characterized by systematic differences in syntax, phonology, grammar, and morphology.

1.4 Types of Dialect

Dialects are classified into two types, but it should be noted that they are inextricably linked [5]:

1.4.1 Geographical dialect

Geographic dialect describes the language variances that occur in different areas or locales as a result of migrations, political boundaries, historical events, or physical distance. Due to these deferences, adjacent localities' speech patterns, vocabularies, and pronunciations differ.

1.4.2 Social dialect

A social dialect indicates the variations in language use influenced by social factors, including social classes and levels of education.

In many cases, members of higher social classes and highly educated individuals tend to use features closer to the standard language. On the other hand, those from lower educational backgrounds preserve more aspects of the native regional dialect.

1.5 Characteristics of Algerian Dialect

Algerian Arabic is the language spoken in Algeria that is distinguished by a lack of standardization and writing resources. It is regarded as a language with insufficient resources. It is not the same as Modern Standard Arabic in any aspect of linguistic representation, including morphology, syntax, lexicon, and phonology.

In this section, we present their most important characteristics [6]:

1.5.1 Lexical characteristics

Like other Arabic dialects, the Algerian dialect has been impacted over centuries by languages such as Berber, Spanish, French, Italian, and Turkish. As shown with some examples in Table 1.1.

Words	Transliteration	Translation	Origin language
فكرون	Fakruwn	A tortoise	Berber
شوربة	Shourbaħ	Soup	Turkish
انتيكة	Antikaħ	Ancient	Italian
صندالة	Sandalaħ	Sandal	Spanish
كسكيطة	Kaskitaħ	Casquette	French

Table *1.1* – The origin and the meaning of some borrowed words used in ALG.

1.5.2 Morphological characteristics

Algerian dialect also has different morphological aspects, consisting essentially of a simplification of some inflections and the inclusion of new clitics as follows:

- The casual endings in nouns and verbs mood are lost;
- The indicative mood is the one that is used as default unlike the other moods that are not used;
- The dual and the feminine plural vanished. And they have been absorbed into the masculine plural;

- The first and second person of the singular form are conjugated in the same way, for example, شْكَرْتْ škart ‘I/you thank’;

1.5.3 Orthographical characteristics

There are two causes for the orthographic variance in Arabic dialect word writing [7]:
-The absence of an orthographic standard due to the lack of codification and standardization of Arabic dialects.
-The phonetic distinctions between the Algerian dialect (ALG) and MSA.

1.6 Challenges in Dialectal Sentiment Analysis

With the increase in interaction with social networking sites in recent years, it has become a rich source full of comments to search for opinions, but these comments carry many challenges, especially concerning preprocessing. This is because the text used in social media usually contains unstructured language that is characterized by not containing capital letters that are usually used to identify features. It also contains many spelling errors, slang words, and abbreviations, in addition to the tendency of many people to repeat letters in some words to exaggerate [7].

One of the other challenges is that there is a lack of an Algerian dialect open-source data set, which led many researchers to collect data and annotate it themselves to advance their research. TWIFIL is one of the open-source Algerian dialect data sets, containing 9000 annotated tweets collected between 2015 and 2019, obtained from different geo-locations in Algeria with the help of 26 annotators [8].

1.7 Techniques for Dialectal Sentiment Analysis

Based on many previous studies, there are three types of sentiment analysis approaches that we mention below, Lexicon-Based approaches, Machine Learning approaches, and Hybrid approaches [9, 10].

1.7.1 Lexicon-Based Approaches

It’s based on counting the number of positive and negative words in text using a predefined opinion lexicon (a predefined list of words). There are two types to build an opinion lexicon [13]:

- **Dictionary-Based Approach:** This approach aims to create a small list of words (positive or negative) collected manually, which will be extended by finding synonyms and antonyms from sources like Wordnet or Thesaurus.

The effectiveness of this method depends strongly on the dictionary size; as the dictionary grows, it becomes exposed to errors.

- **Corpus-Based Approach:** The corpus-based approach relies on large collections of text for syntactic and semantic patterns of opinion**.** To work effectively, this method often needs a massive dataset with labels.

1.7.2 Machine learning approaches

Machine learning is a branch of artificial intelligence that aims to allow computers to learn without being explicitly programmed for specific tasks by developing different algorithm types [11].

The Machine Learning-based technique can be divided into supervised, unsupervised, semi-supervised, and reinforcement learning [12].

1.7.2.1 Supervised learning approach

A supervised approach necessitates labeled documents, typically with positive, negative, and neutral classes. This approach consists of four main types: probabilistic, linear, decision tree, and rule-based [13].

• **Linear approach:** a linear approach in statistics refers to a method of sentiment classification that utilizes linear or hyperplane decision boundaries.

- **Support Vector Machine (SVM):** stands out as one of the most common classifiers in machine learning for pattern classification, which was presented by Vapnik in 1995. Using a variety of kernel functions, its main goal is to convert nonlinear, separable data points into a higher-dimensional space [14].
- **Artificial Neural Network (ANN):** recently, neural networks have become a crucial classification tool, offering a compelling alternative to traditional classification techniques [15]. As commonly understood, an artificial neural network comprises units known as neurons inspired by the structure and behavior of natural neurons.

  The Neural network architecture includes three layers: an input layer, a hidden layer, and an output layer. The first accepts input data, while the second could be single or multiple layers positioned between the input and output layer, the final layer namely the output layer typically contains a single neuron that generates results between 0 and 1 using activation functions [16].

• **Probabilistic approach:** The probabilistic approach includes naïve Bayes, Bayesian Network, and Maximum Entropy models, this approach classifies text into sentiment categories in addition to quantifying the associated uncertainty.

- **Naïve Bayes (NB):** Naive Bayesian is a simple Classification method commonly used in sentiment analysis tasks, based on the probability theorem which was presented by Thomas Bayes, it's called "naive" because it assumes that the characteristics it uses for classification are independent [17].

Bayes' Theorem is expressed mathematically as shown in equation (1):

$$P(C|X) = \frac{P(c)*P(x|c)}{P(x)} \qquad (1)$$

Where: P(c|x) is the probability of class c given the data x, P(c) is the prior probability of class c, P(x|c) is the probability of observing data x given that class c is true, and P(x) is the probability of observing data x.

- **Bayesian Network (BN):** Judea Pearl introduced Bayesian networks (bns) as a probability distribution of a set of discrete random variables presented graphically. To put it simply, BN is a flowchart representation of nodes, and arrows, So that the nodes represent random variables, while the arrows represent probabilistic relationships between them. These networks are really helpful for classification tasks by offering structured ways to model and reason about uncertainties in classification problems and calculating the probability of different classes given the input data [18].
- **Maximum Entropy (ME):** also recognized as a conditional exponential classifier or Maxent classifier, it estimates the conditional distribution of the class label c given a document d to maximize the system's entropy instead of operating with assumptions about feature relationships. To determine this, we use the exponential form as shown in equation (2):

$$P_{ME}(c|d) = \frac{1}{Z(d)} exp\left(\sum_i \Lambda_{i,c} F_{i,c}(d|c)\right) \qquad (2)$$

Where Z(d) is a normalization function, $F_{i,c}$ Represents a feature function for feature F_i, and class c, and $\Lambda_{i,c}$ Denotes a parameter ensuring observed features match expected ones within a given set [12].

• **Rule-Based Approach:** in the Rule-based approach each record or instance of data is classified by employing a set of specific rules, these rules can be typically ordered or unordered, and each one contains a condition and a consequent to predict the target class. The output of a rule-based sentiment analysis classifier could be applied to various domains, such as movie reviews, and product feedback [19].

• **Decision Tree Approach:** A decision tree is a hierarchical machine learning model that establishes decisions based on specific attributes of the Input data. Basically, it’s a flowchart that starts with a root node and branches out into terminal nodes, each one represents a decision based on a particular attribute. Until reaching the terminal node these decisions lead to other branches, which provide the final output [20].

• **K-NN-Based Approach:** The K-Nearest Neighbors is a multi-task classification technique that aims to assign labels to data points based on other labeled data points.
Due to his performance, and simplicity, he’s widely used in the sentiment classification field.
It’s named ”K-Nearest” because it classifies new data points based on measures of the closest distance between several labeled data chosen (k), using typically Euclidean distance [21].

1.7.2.2 Unsupervised learning approach

Unlike supervised learning, this approach Doesn't necessitate labeled data which is a good advantage due to the difficulty of gathering a labeled document for text classification tasks, but it should also be noted that the approach normally needs a huge amount of data [22]. This approach is based on the clustering technique which can be categorized into two main groups namely hierarchical clustering and partitional clustering algorithms [23].

• **Hierarchical clustering algorithm:** It’s a method of cluster analysis that builds a hierarchy of clusters. It starts with each data point as its own cluster and iteratively merges or splits clusters based on their similarity, creating a tree-like structure known as a dendrogram. These algorithms can be Partitioned as divisive (top-down) and agglomerative (bottom-up) [23].

• **Partitional clustering algorithm:** these algorithms portioned the data into un-nested groups in such a way that the data of one cluster have the most similarities, one of the most popular clustering algorithms called K-means [23].

1.7.2.3 Semi-supervised learning approach

It is a type of machine learning approach that aims to combine supervised and unsupervised learning.

The main objective of this approach is to avoid the difficulties of gathering a large labeled amount manually of data by training a dataset containing both labeled and unlabeled data.

The semi-supervised learning is subcategorized into two types: Semi-supervised classification which predicts the class labels for unlabeled data points, and Semi-Supervised Clustering to partitions a dataset into groups or clusters [24].

1.7.2.4 Reinforcement learning approach

The reinforcement learning approach is completely different compared to supervised and unsupervised techniques, RL is often used to solve problems by making a sequence of decisions.

It is crucial to note that the RL is characterized by an agent interacting with an environment to learn optimal behavior by making a series of trials and errors.

There are three main categories of RL: dynamic programming, Monte Carlo methods, and temporal difference methods. Each one of them has its strengths and weaknesses and is applicable in different scenarios [25].

1.7.2.5 Deep Learning-Based approach

Deep learning-based approaches encompass numerous techniques inspired by Artificial Neural Networks (ANN), with the most prominent being Recurrent Neural Networks (RNN), Long Short-Term Memory Networks (LSTM), and Convolutional Neural Networks (CNN). These models learn from labeled data by adjusting weights over multiple epochs. [3].

- **Recurrent Neural Network (RNN):** RNNs are considered promising tools for machine translation tasks. Unlike feed-forward neural networks, RNNs are capable of processing sequences of varying lengths by incorporating a recurrent hidden state. This state's activation at each time step relies on the previous time step's activation (see Figure 1.1). The computational operations within an RNN follow specific formulas as shown below in equation (3) [26]:

$$h_t = \sigma(U\,x_t + W\,h_{t-1}) \tag{3}$$

Where x_t Represents the input at time t, while h_t Represents the hidden state and the memory of the network at time step t.
It is crucial to note that h_t Initialized to a zero vector.

$$o_t = softmax(V\,h_t) \tag{4}$$

While the o_t Parameter represents the output at the time step, is computed using the softmax function as shown in Equation (4), in the case of sentiment classification this output is typically a vector containing probabilities for each sentiment category.

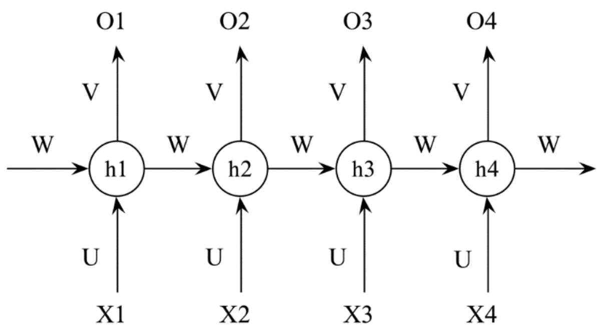


Figure *1.1* – An unrolled recurrent neural network [26].

- **Long Short-Term Memory:** LSTM is a type of recurrent neural network (RNN) Designed to solve the problem of vanishing or exploding gradients encountered by traditional RNNs. LSTM like other RNNs, to generate their output they use information from the current time-step as well as the output from the previous time-step and then pass it to the next time-step. Each LSTM unit contains a memory cell (denoted as c_t) that maintains its state over different time intervals, along with three non-linear gates: an input gate (i_t), a forget gate (f_t), and an output gate (o_t) as shown in figure 1.2 bellow. These gates are addressed to control the flow of information into and out of the memory cell, by applying mathematical functions such as the sigmoid function, the hyperbolic tangent function, and element-wise multiplication.

  It is important to note that (x_t) represents the input vector, (h_t) represents the hidden state vector at time t, U and W represent weight matrices for the gates or memory cell, and b represents bias vectors.

  In addition, The LSTM transition continued based on the equations (5 - 10) given below [27]:

$$f_t = \sigma\left(w_f h_{t-1} + u_f x_t + b_f\right) \tag{5}$$

$$i_t = \sigma(w_i h_{t-1} + u_i x_t + b_i) \tag{6}$$

$$C'_t = \mathrm{Tanh}(w_c h_{t-1} + u_c x_t + b_c) \tag{7}$$

$$c_t = f_t \,.\, c_{t-1} + i_t \,.\, c'_t \tag{8}$$

$$o_t = \sigma(w_0 h_{t-1} + u_0 x_t + b_0) \tag{9}$$

$$h_t = o_t \,.\, Tanh(c_t) \tag{10}$$

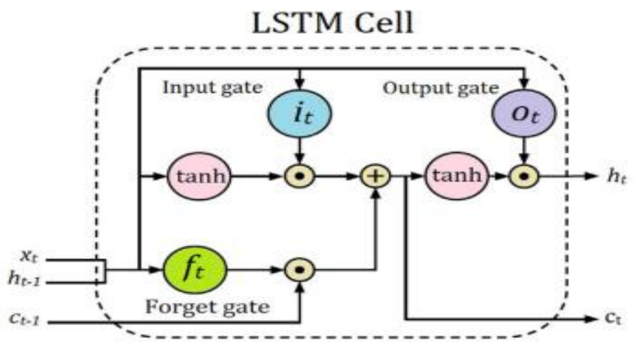


Figure *1.2* – LSTM Architecture [27].

- **Convolutional Neural Network:** Put simply, cnns is considered a type of neural network generally utilized in natural language processing (NLP) tasks for extracting local features [27].

  A standard CNN comprises an input layer, convolutional layers, pooling layers, a fully connected layer, and an output layer, as illustrated in Figure 1.3.

  And employed the gradient descent algorithm for parameter optimization and adjustments where a learning rate η and λ parameter is used to control the overfitting [28].

  When we suppose h_i Is a convolutional layer, as described in equation (11) below:

$$h_i = f(h_{i-1} . w_i + b_i) \qquad (11)$$

  Where:

  w_i Is the weight vector of (h_i), “.” Is the convolution operator of the kernel.

  Whether, if the h_i Is considered as a pooling layer to reduce dimensionality and keep the features stable, it can be expressed as shown in equation (12) below:

$$h_i = subsampling\ (h_{i-1}) \qquad (12)$$

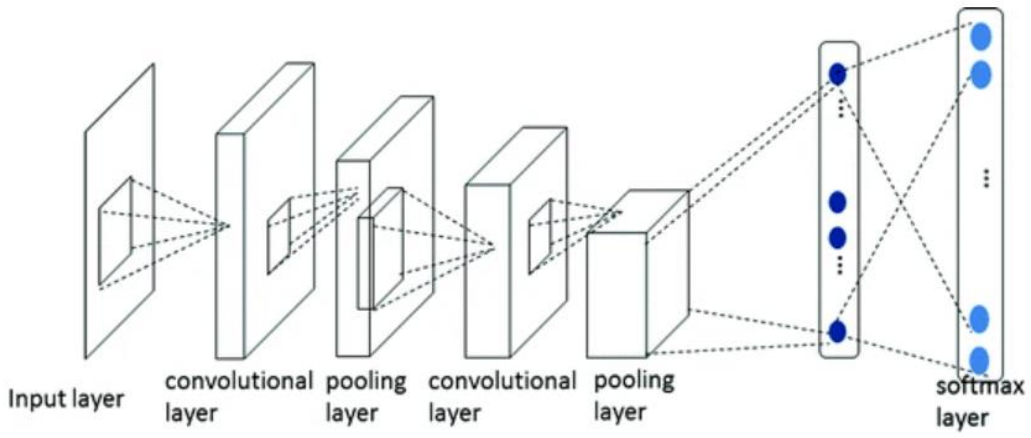


Figure *1.3* – Structure of typical CNN [28].

- **Transformers:** Transformers are a type of advanced machine learning model designed especially for sequence transduction tasks, such as language translation and text generation, these models use an attention mechanism and sequence-to-sequence architecture, enabling them to capture complex contextual information effectively.
  The architecture of transformers consists of an encoder-decoder, as shown in Figure 1.4.

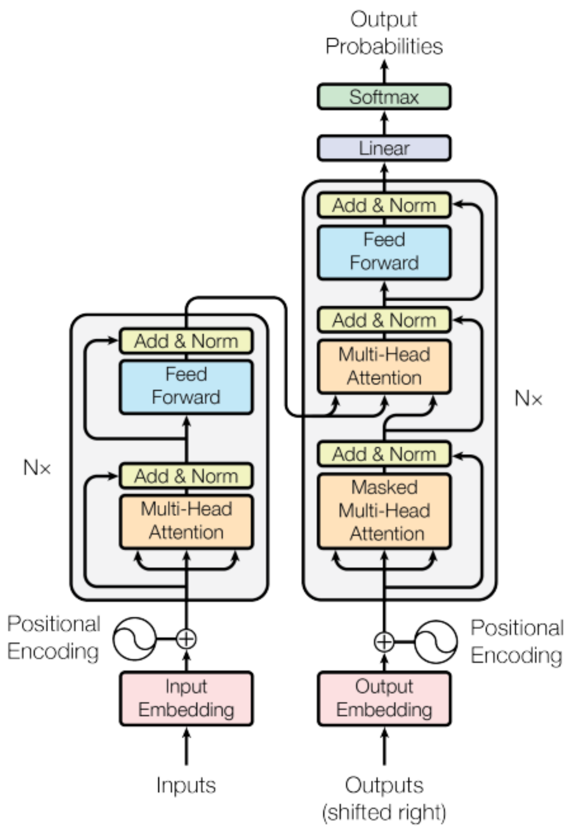


Figure 1.4 – The Transformer model architecture [29].

The Encoder consists of a multi-head self-attention mechanism followed by a feedforward neural network. These layers allow the model to capture hierarchical representations and process the input sequence, while the decoder comprises a multi-head self-attention mechanism, followed by encoder-decoder attention and a feedforward neural network. The decoder layers enable the model to generate the output sequence based on the context provided by the input sequence [29].

There are many transformer models, we mention among them BERT (Bidirectional Encoder Representations from Transformers) which is a powerful Natural Language Processing (NLP) model developed by Google Research in 2018. BERT is based on an encoder stack of transformer architecture, which uses self-attention to understand the context of words in a sentence.

BERT training involves two main steps: pre-training and fine-tuning. During pre-training, the model learns from unlabeled data through various tasks. In fine-tuning, BERT's pre-trained parameters are adjusted using labeled data specific to the downstream tasks it's applied to. Each task may require its own fine-tuned model, even though they all start with similar pre-trained parameters. This approach allows BERT to adapt effectively to diverse NLP tasks with high accuracy [3].

Within the realm of BERT models, several variants have been meticulously trained on extensive datasets. Notable among these are:

• **Arabic BERT**: refers to a collection of BERT language models specifically designed for the Arabic language. These models were developed from scratch and are available for public use. The collection, known as arabicbert3, comes in various model sizes to accommodate different needs for computational resources and performance. BERT-Mini has 4 hidden layers, 4 attention heads, a hidden size of 256, and 11 million parameters. BERT-Medium increases to 8 hidden layers, 8 attention heads, a hidden size of 512, and 42 million parameters. BERT-Base further expands to 12 hidden layers, 12 attention heads, a hidden size of 768, and 110 million parameters. Finally, BERT-Large offers the most complexity with 24 hidden layers, 16 attention heads, a hidden size of 1024, and 340 million parameters.

These models were trained using masked language modeling with whole-word masking techniques.

The training data for arabicbert3 comprised a combination of the unshuffled version of OSCAR data and a recent data dump from Wikipedia, totaling approximately 8.2 billion words. To construct the vocabulary set, 32,000 Wordpieces were utilized. It's worth noting that the Arabic characters do not have upper or lower case distinctions, so there is no distinction between cased and uncased versions of the model [30].

• **Distil Bert:** is a refined iteration of the BERT base multilingual model, achieved through a process of distillation. Unlike its predecessor, this model maintains case sensitivity, distinguishing between English and other forms of language.

Trained on a conglomerate of Wikipedia content spanning 104 different languages, distilbert is characterized by its condensed architecture featuring 6 layers, each with 768 dimensions and 12 heads, amounting to a total of 134 million parameters. This streamlined design results in enhanced operational efficiency, with distilbert demonstrating an approximately twofold increase in speed compared to other Bert architectures [31].

• **aragpt2:** aragpt-2 is a family of Arabic language models optimized for text generation, These models leverage the powerful architecture of a Generative Pre-training Transformer (GPT) but come in various sizes and complexities to cater to different needs. Aragpt2-base uses the lamb optimizer, has a context size of 1024, an embedding size of 768, 12 attention heads, 12 layers, and comprises 135 million parameters in a 527MB model. Aragpt2-medium, also using the lamb optimizer, features the same context size, an embedding size of 1024, 16 attention heads, 24 layers, and 370 million parameters in a 1.38GB model. Aragpt2-large switches to the adafactor optimizer, maintaining the context size, with an embedding size of 1280, 20 attention heads, 36 layers, and includes 792 million parameters in a 2.98GB model. Aragpt2-mega, also using adafactor, has a context size of 1024, an embedding size of 1536, 25 attention heads, 48 layers, and boasts 1.46 billion parameters in a 5.5GB model. The dataset used to train these models consists of 77GB of text, encompassing 200,095,961 lines, 8,655,948,860 words, and 82,232,988,358 characters [32].

- **Word Embedding:** a vector space representation that groups words with related meanings together, each word gets its own set of numbers, which are learned in a way similar to how computers learn patterns. These sets of numbers represent the words in a special mathematical space.

  There are different methods to create these word embeddings. Word2vec is a popular one. It's like a simple two-layer computer program that turns words into numbers. It has two modes: one predicts a word from its neighbors, while the other predicts neighbors from a word.

Another method is Global Vectors (glove). It's like a big puzzle game where the computer determines how words fit together based on how often they hang out in the same sentences.

Fasttext is a tool made by Facebook that's like a super-fast version of word2vec. It's great for understanding the meanings of words in different contexts and finding words that go well together [33].

1.7.2.6 Ensemble Learning Approaches

Ensemble learning is a machine learning paradigm where multiple models, often referred to as "weak learners," are trained and combined to solve a particular computational intelligence problem. The primary goal of ensemble learning is to improve the performance, robustness, and accuracy of the model by leveraging the strengths of different learners and compensating for their individual weaknesses [34].

Ensemble learning methods like bagging, boosting, voting, stacking, blending, and cascading improve model performance by combining multiple algorithms to reduce errors, improve robustness, and mitigate overfitting. In our project, we employed two notable methods: voting and stacking [35].

•**Voting:** is a simple ensemble method where multiple models are trained independently and their predictions are combined using some form of averaging (hard voting or majority voting for classification, and soft voting or weighted averaging for regression).

•**Stacking (Stacked Generalization):** involves training a meta-learner to combine the predictions of several base learners. The base learners are trained on the original dataset, and their predictions are then used as input features for the meta-learner, which makes the final prediction.

1.7.3 Hybrid Approaches

In order to enhance the performance and handle the ambiguous language in sentiment analysis classification, the hybrid approach combines the speed of Lexicon-based techniques with the adaptability of machine learning. By blending techniques from both approaches, the hybrid method addresses their respective limitations and leverages their advantages. It does this by using scores generated from lexicon analysis as features for the sentiment classifier. Consequently, sentiment dictionaries play a crucial role in the hybrid approach, which is recognized for achieving superior performance compared to either method used alone [12].

Finally, we have a simple comparison of different approaches in sentiment analysis shown in the table below [19].

Methods	**Advantage**	**Limitation**
Machine Learning	• Eliminate the need for a dictionary. • Achieve high precision in classification. • Ensure high accuracy and flexibility.	• More time is needed. • It varies depending on the field. • Human participation and information labeling are necessary.
Lexicon Based	• Doesn't require labeled information. • varies depending on the domain. • demands less time.	• Requires extensive linguistic tools. • Low precision. • Requires dictionaries encompassing various perspectives.
Hybrid Method	• Achieves efficiency with reduced time requirements. • Leverages the combined strengths of various processes. • Enhances accuracy and precision.	• Low precision. • Reliability is lacking.

Table 1.2 – Comparison of various Sentiment Analysis approaches [19].

1.8 Related works

We can categorize several studies related to the Algerian Arabic sentiment analysis field into three cases, which we mention below [36].

1.8.1 Case of Arabic Algerian dialect (Latin or Arabic script)

In exploring sentiment analysis within Algerian dialect text, whether in Latin or Arabic script, In [36] authors stand out employing a Bert base Transformer model architecture with 12 encoders, 12 attention heads, and a hidden dimension of 768. The study commenced with collecting approximately a million tweets from different parts of Algeria Through the use of the Twitter API, then there was a training of a wordpiece Tokenizer on the amassed dataset, comprising a vocabulary of 50,000 words. Subsequently, the language model underwent training utilizing the Masked Language Modeling (MLM) task, with a 25% probability and a batch size of 64 due to the limitations of the computational resources, spanning nearly 10 days across 50 epochs. The results obtained are very promising compared to other models. For both data sets, Narabizi and Twifil (63.5% accuracy for Narabizi, and 80.5 accuracy for Twifil). To ensure convenient access in the future, the completed pytorch version named dziribert has been uploaded to the Transformers Hub.

Another recent work realized by [37], contains many stages to perform sentiment analysis on Algerian dialects written in both Arabic and Latin characters.

In the first stage, 45,000 comments were extracted using YouTube API from 30 Algerian channels, after that, the collected comments were labeled manually into 3 classes positive, negative, or neutral. Sequentially, many data preprocessing tasks have been realized to get better performances including stop word removal, stemming, and Normalizing special Arabic characters. After the preprocessing stage, they fine-tuned many Bert models with different architectures trained already on a huge amount of Arabic MSA data in addition to two deep learning models (LSTM and bidirectional LSTM), for the parameters of training the best combination obtained is a 16 batch size, 180 sequence length, 800 as the number of warm-up steps. And $1.07 \times 10-5$ learning rate which is very low to avoid early over-feating. Finally, after evaluating the model performance they chose between stopping the procedure and continuing to get some improvements by adding or removing preprocessing tasks after training and evaluation, The achieved final results are promising, showcasing the superior performance of the large BERT model. It attained an impressive accuracy of 0.81 and an F1 score of 0.78. The best model chosen was uploaded to the Huggingface page for further use in the future.

In Ref. [38] authors introduced an alternative approach to address the diverse text preprocessing difficulties arising from different forms of comment writing in Modern Standard Arabic (MSA), French, or local dialect. Their methodology involves several stages to handle this complexity. Initially, they collected approximately 11,760 comments from various social media platforms, including Facebook, Twitter, and YouTube. A filtering operation followed, eliminating comments written in other languages and removing repeated comments. After annotating the data into two classes, positive and negative, they applied preprocessing steps such as tokenization, normalization, removing stop words, transcribing Arabic characters to Latin, and grouping.

Sequentially, From 466,424 texts (31.9 MB), they constructed two word2vec models (CBOW and Skip-Gram with k = 50, determined experimentally) to enrich the dataset by adding similar words to the original comments. The training process involved applying both machine learning and deep learning classification methodologies to the original and the transcribed datasets. The best results for both datasets were achieved using SVM, with the original dataset reaching an accuracy of 72.04% and the transcribed dataset achieving 84.21% accuracy. Among the deep learning models, RNN provided the best results on the transcribed dataset with 65.21% accuracy, while CNN performed best on the original dataset with 62.96% accuracy.

On the other hand, the research conducted by [39] covers binary sentiment analysis of Modern Standard Arabic (MSA) and the Algerian dialect across three phases. In the data collection phase, two sets of data were gathered. The first dataset, used for sentiment analysis model training, comprised 100,000 comments extracted from popular Algerian Facebook pages, along with existing open-source datasets for both MSA and the Algerian dialect. After annotating and balancing, this dataset included 49,864 positive comments and 49,864 negative comments. The second dataset, aimed at training the word2vec model (CBOW), combined five corpora totaling 1.4 gigabytes in size. In the data preprocessing phase, non-Arabic content was deleted and Arabic characters were normalized. In the final phase, two models were trained: the SVM model and the LSTM model. The dataset was split into 85% for training and 15% for testing. The SVM model was trained using the TF-IDF technique with parameters set to a minimum document frequency of 5 and a maximum document frequency of 0.95, while the LSTM model used the CBOW model from the gensim library for word embedding, with a batch size of 20, 64 LSTM units, and a maximum sequence length of 250. After training, the SVM model achieved an accuracy of 0.86, and the LSTM model achieved an accuracy of 0.81.

Another work proposed by [40], tackled the sentiment analysis of Algerian comments written in Arabic or Latin characters.

The proposed approach comprises six main processes: Initially, data collection involved utilizing an established dataset called "Wach-t7ass," created in 2016 by [67]. In addition, comments were extracted from the Facebook platform using Facepager and the Facebook graph API. Moreover, comments along with their respective classes were collected through Google Forms. Subsequently, the annotation stage was achieved by classifying the data into positive, negative, and neutral classes using the open-source platform ”DOCCANO.” Following annotation, many preprocessing steps were undertaken to refine the data, such as tokenization, noise removal, vowel removal, stop words removal, treatment of capitalized words, deduplication of comments, and removal of tags and URLs, in addition to applying transliteration to comments written in Latin characters. Afterward, the cleaned data was presented using three vector types: count-vectorizer, TF-vectorizer, and TF-IDF vectorizer. Finally, in the classification phase, three algorithms were employed: Naive Bayes, Support Vector Machine, and Decision Tree.

In tests and experiments, the preprocessing steps notably exhibit a significant influence on training performances, and the SVM outperforms other algorithms and returns the best scores by reaching 83.28% accuracy.

In 2018, Guellil and his team presented another work on Algerian dialectal sentiment analysis employing an automated corpus annotation approach [41]. Their contribution consists of three general stages.

Firstly, the DALG Sentiment Lexicon was automatically constructed by assigning the scores of similar translated words in the English Lexicon.

Then 7,926,504 messages in Arabic and 3,976,700 in Arabizi were collected through the restfb API. These Facebook messages underwent automated labeling, applying several key treatments, including negation treatment and pretreatment of messages, to standardize their format and remove noises.

Finally, for the classification phase, several classifiers were employed on both datasets (Arabic and Arabizi), such as support vector machine, naive Bayes, logistic regression, decision tree, and random forest, utilizing two vectorization techniques: bag of words (BOW) and document embedding. Based on simulations and analysis, we notice that the results in the Arabic corpus surpassed those in the Arabizi corpus. Notably, the BOW vector representation yielded superior performance, particularly in logistic regression, achieving an accuracy of 0.78 and an F1 score of 0.78.

1.8.2 Case of Modern Standard Arabic

In the realm of contemporary Modern Standard Arabic sentiment analysis, we cite Ref. [42]. In this work, two distinct corpora were used. The first corpus, OCA, is already available and freely accessible, while the second corpus, SANA, comprises 718 comments collected from three Algerian newspapers and annotated by two Algerian Arabic native speakers, with annotation agreement measured using the kappa coefficient. Sequentially, a preprocessing step was applied, including translating comments in French, English, or Arabizi into MSA, tokenization, stemming, stop word removal, and defining n-grams and word vectors using methods such as TF, TO, TF-IDF, and BTO. For training, they employed SVM with a linear kernel, Naïve Bayes, and k-nearest neighbors (k=9), using 10-fold cross-validation to prevent overfitting. The best accuracy for the OCA corpus was 89% with Naïve Bayes using Tri-gram and TO encoding, while the SANA corpus achieved 75% accuracy with Naïve Bayes using Bigram and TF encoding. A notable finding was that light stemming's benefits varied based on the corpus characteristics.

1.8.3 Case of Arabic Algerian dialect (Arabic Script)

In 2018, Soumeura and other researchers addressed significant challenges in sentiment analysis of Algerian dialect comments [43]. Their approach included four main stages. First, they collected approximately 100,000 comments from over 20 Algerian brand pages using the Facebook API. These comments were manually annotated and classified into positive, negative, and neutral categories by two native annotators, with a third annotator resolving disagreements, resulting in 25,475 annotated comments. The data was then preprocessed to remove repeated letters, diacritics, elongation, and stop words. Emojis were categorized into six types, each replaced by its category name. French comments were identified and translated into Modern Standard Arabic (MSA) using the Google Translate API, while Latin-letter comments were transliterated into Arabic. For classification, they implemented a Naive Bayes model, a Multi-Layer Perceptron (MLP), and a Convolutional Neural Network (CNN) with three hidden layers. The CNN achieved an impressive accuracy of 89.5%, surpassing the MLP's 81.6% accuracy.

In 2021, Imane Guellil and other researchers introduced an innovative approach to sentiment analysis for Algerian dialect comments using both automatically constructed and manually reviewed corpora [44]. The approach involved extracting comments from 226 popular Algerian Facebook pages and using the Facebook Rest API with MSA/DALG keywords, resulting in 7,926,504 comments after preprocessing steps such as removing repeated messages, special characters, and Tatweel characters. Arabizi messages were transliterated into Arabic script through specific rules and a machine translation system was built for translation between Arabizi and MSA. An automated corpus lexicon construction methodology, using the Glosbe API, translated an English lexicon to DALG and MSA, yielding 1,745 curated sentiment terms. For classification, a comparison between shallow algorithms (e.g., gaussiannb, logisticregression, randomforest) and deep learning models (e.g., CNN, LSTM, Bi-LSTM) was conducted, utilizing Word2Vec and fasttext for feature extraction. The best machine learning model achieved an F1 score of 0.89 on the Test_sentialg corpus, while CNN with fasttext also attained an F1 score of 0.89, with better results observed on manually reviewed corpora.

We present in table 1.3 a comprehensive overview of previous research in the realm of sentiment analysis in the Algerian dialect.

Ref	Authors	Year	Dataset	Approaches	Results
[36]	Amine Abdaoui	2022	Over one million tweets	Dziribert: a Pre-trained Bert base Model adopted for the Algerian Dialect	63.5% on Narabizi, 80.5% on Twifil
[37]	Zakaria Benmounah	2024	Algerian youtube comments (45.000)	(Fine-tune Bert Arabic model variants, Lstm, Bi-Lstm)	81% accuracy with Bert Arabic large variant
[38]	Ahmed cherif mazari, Abdelhamid djeffal	2022	11760 comments collected from diverse social media platforms	(BNB, MNB , GNB, LSTM , CNN, RNN)	Models without word2vec (MNB:84.21%, CNN:64.11%) Models enhanced by word2vec accuracies (SVM:83.70%, RNN:65.21%).
[39]	Adel Abdelli, Fayçal Guerrouf, Okba Tibermacine, and Belkacem Abdelli	2019	49864 comments extracted from popular Algerian Facebook pages	(Lstm, Svm)	Accuracies (Svm: 86%, Lstm: 81%)
[40]	Asma Chader, Dihia Lanasri, Leila Hamdad,	2019	(“Wach-t7ass” dataset + comment	(Naive Bayes, SVM, Decision trees)	Accuracy 83% For SVM

	Mohamed Chemes Eddine Belkheir, and Wassim Hennoune		extracted from Facebook + comments collected through Google Forms)		
[41]	Imane Guellil, Ahsan Adeel, Faical Azouaou, and Amir Hussain	2018	Facebook messages	(Naive Bayes, SVM, Decision trees, Logistic regression, Random forest)	Accuracy: O.78 F1 score: 0.78
[42]	Hichem Rahab, Abdelhafid Zitouni, and Mahieddine Djoudi	2021	OCA (freely accessible) , SANA (comprises 718 Algerian newspaper comments)	(SVM, Naïve Bayes, K-nearest neibors)	NB classifier(OCA: 89% Accuracy SANA: 75% Accuracy)
[43]	Assia Soumeura, Mheni Mokdadia, Ahmed Guessouma, and Amina Daoudb	2018	25475 comments extracted from Facebook pages	(Multi-layer perceptron, CNN, Naive Bayes)	Accuracies (CNN: 89.5%, MLP: 81.6%)
[44]	Imane Guellil and other researchers	2021	Facebook comments(Corpus constructed automatically +corpus reviewed manually)	(gaussian nb, Logistic Regression, random forest, sgd classifer, CNN, LSTM, and Bi-LSTM)	F1 scores(SGD: 89%, CNN: 89%)

[45]	Khaoula Hamadouche, Kheira zineb Bousmaha, Bekkoucha Mohammed Abdelwaret, and Lamia Hadrich Belguith	2023	54000 comments collected from social networking (Twitter, youtube)	Algbert: Automatic Construction of Annotated Corpus for Sentiment Analysis in Algerian Dialect	91.04% accuracy
[46]	Nawapon Kewsuwun and Siriwan Kajornkasirat	2012	1001 Facebook comments in Thai language	A sentiment analysis model of agritech startup on Facebook comments using naive Bayes classifier	Fmeasure:75%, precision: 80%, recall: 75%, Accuracy: 61%

Table 1.3 – realized works in Sentiment Analysis for the Algerian dialect.

The works referenced in Table 1.3, provide a rich overview of sentiment analysis efforts across various Algerian dialect datasets using different machine learning and deep learning approaches. However, they exhibit several limitations. Firstly, there's a variability in dataset sizes and sources, ranging from a few thousand to over a million comments, which might affect the generalizability of the models. The approaches used also differ significantly, from traditional machine learning techniques like Naive Bayes and SVM to more advanced methods such as BERT variants and deep neural networks. This inconsistency makes direct comparisons challenging. Additionally, the models show varied performance metrics, often influenced by the specific dataset and preprocessing techniques employed.

Another limitation is the relatively narrow focus on certain social media platforms, which may not fully capture the diversity of the Algerian dialect. The use of high computational resources required for training and fine-tuning BERT models, such as dziribert and other large transformers, is a significant constraint. These models demand powerful hardware (e.g., gpus or tpus) and substantial memory, making them less accessible for researchers with limited resources.

Moreover, the problem of annotation quality is a critical issue. High-quality labeled data is essential for training accurate sentiment analysis models, but manual annotation can be inconsistent and subjective, leading to noisy labels. This is particularly challenging for Algerian dialects, where linguistic nuances and variations are prevalent. Inconsistent or poor-quality annotations can significantly impact the performance and reliability of the models.

Lastly, some studies report high accuracies without extensive validation on unseen data, potentially indicating overfitting. Thus, while these works collectively advance sentiment analysis in Algerian dialects, they highlight the need for standardized datasets, consistent evaluation metrics, broader data sources, and consideration of computational resource constraints and annotation quality to improve comparability and robustness.

1.9 Conclusion

In conclusion, this chapter explored the vast field of sentiment analysis with a specific focus on Algerian dialects. Through an overview of sentiment variations and the most frequent techniques employed in this domain, combined with a profound examination of dialects, their types, and their characteristics.

This survey empowered us to gain insight into the complexities of analyzing sentiment in numerous linguistic contexts.

The broaching of challenges in dialectal sentiment analysis highlighted the many-sided nature of this task, including issues related to data availability, annotation, linguistic variation, and socio-cultural factors. These challenges underscore the need for innovative approaches and techniques adapted to the nuances of dialectal sentiment.

While the comparison of techniques for dialectal sentiment analysis in recent works has provided valuable insight into potential methodologies that aim to address these challenges. From machine learning algorithms or deep learning models to linguistic analysis tools, a range of techniques offer promising avenues for advancing research in this field.

Overall, this chapter contributes to the ongoing discourse on sentiment analysis by emphasizing the importance of considering dialectal variations. By acknowledging and addressing the unique characteristics of Algerian dialects, we move towards more inclusive and accurate sentiment analysis frameworks that better reflect the diversity of language and expression. As technology continues to evolve, further research in this area will undoubtedly lead to more sophisticated and effective approaches to understanding sentiment in dialectal contexts.

CHAPTER 2: Methodology

2.1 Introduction

In this chapter, we present our comprehensive approach that includes many stages to perform a comparison of diverse sentiment analysis models on an Algerian Arabic dialect corpus, our comparison is customized to distinguish between positive and negative sentiments across both machine and deep learning models.

Overall, we detail each step of our methodology to illustrate the process thoroughly.

2.2 Proposed approach

Our contribution presented in Figure 2.1 consists of many supervised classifiers either for machine learning or deep learning.

Each of these classifiers has been learned on labeled corpora, enabling them to adeptly predict the sentiment classes of new texts, distinguishing between negative and positive sentiments.

According to machine learning models we have employed SVM with linear kernel, KNN, and Naïve Bayes multinomial classifier, each one used TDF-IDF for text representation and employed the CHI-2 test to select the informative features. As for deep learning models LSTM model, CNN model, and different transformer-based architecture models including BERT and GPT are trained. Where we have trained the word2vec model to encode the data for both LSTM and CNN's models, and we employed the wordpiece tokenizer to present the text in Bert models and the Byte Pair Encoding for the GPT model.

To enhance our transformer models aragpt2, arabicbert base, and distilbert we implemented two ensemble learning techniques: voting and stacking.

In the voting approach, we used majority voting to aggregate the predictions from the individual models. For the stacking approach, we utilized the predictions from these transformers as input features to train a meta-model, specifically using a logistic regression classifier.

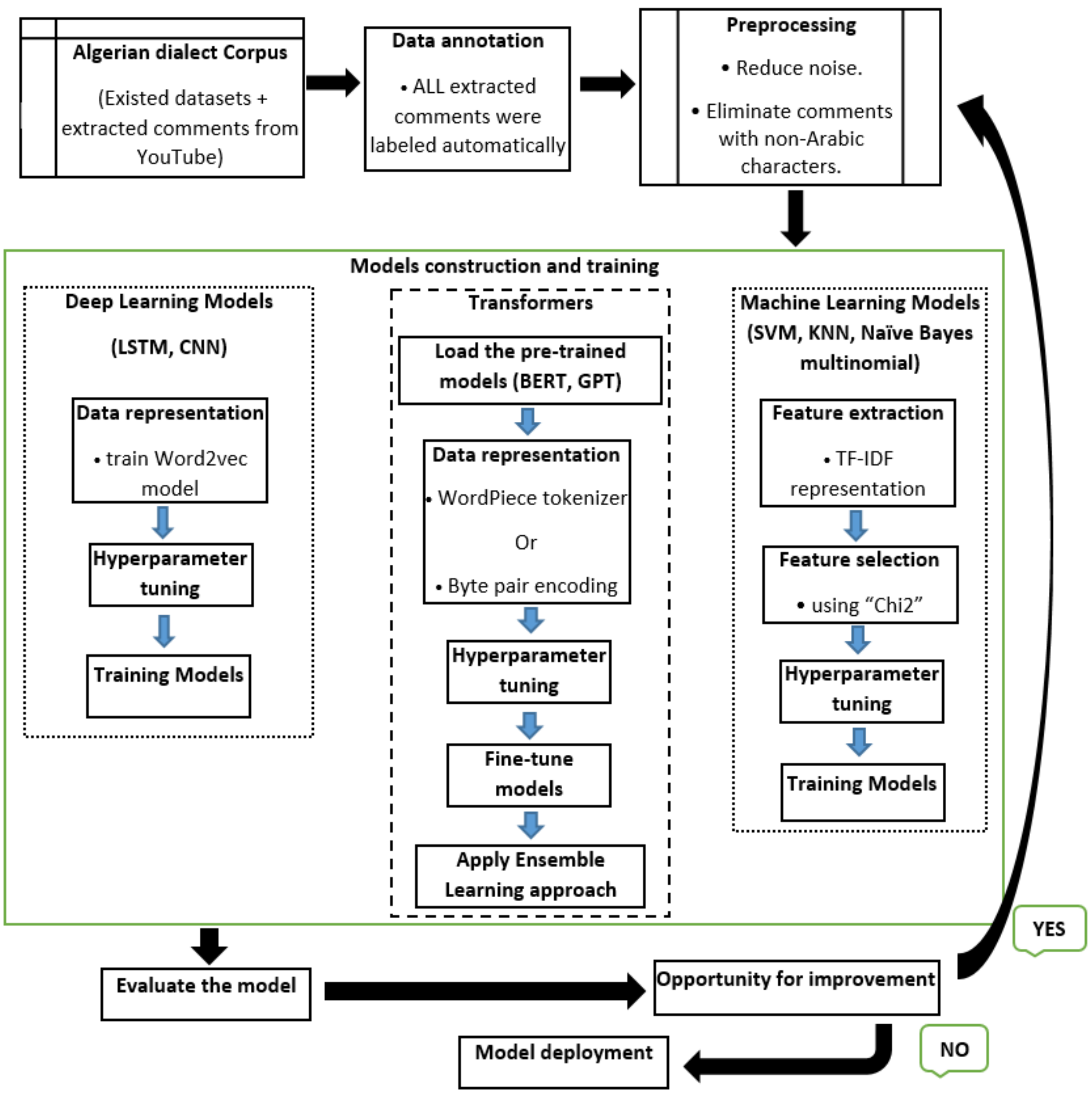


Figure 2.1 – Workflow diagram of the proposed approach.

2.3 Refining the Steps

2.3.1 Data collection

In our experimentation, we have amalgamated four different data sets:

1. Dzsentia dataset: which contains 50.000 comments in the algerian dialect already labeled by abdelli into two classes: positive and negative, extracted from the youtube platform [39].
2. Dziribert dataset: while the complete version of this dataset remains private and encompasses approximately one million tweets, for our study we have used the publicly available part of this dataset which contains 9437 and is posted on the github page of amine abdaoui [36].

3. Another dataset realized by Zakaria Benmounah [37] was merged with the previous datasets, this one comprises 45.000 comments sourced from the youtube platform, categorized into negative, neutral, and positive classes.

In addition, we used youtube API **to gather 8252** comments in the Algerian dialect covering various topics, subsequently, we manually annotated these comments into three classes namely positive, negative, and neutral, Since we are focusing on binary classification, we eliminated the neutral comments, resulting in a dataset of 5,331 comments (3,442 negative and 1,889 positive).

To ensure that our models perform well and generalize effectively, we employed two distinct datasets: TWIFIL [8] and NARABIZI [47], which contain after preprocessing stages 2,380 and 725 comments respectively. By testing the models on these diverse datasets, we aimed to evaluate their robustness and adaptability to different linguistic characteristics and sentiment distributions. This approach helps us identify potential overfitting issues and ensures that the models can handle variations in language and context effectively.

2.3.2 Preprocessing

To prepare our data before its use in our classification models, we have applied several preprocessing tasks that aim to improve the performance of our classifiers by removing any irrelevant or noisy data. After the experimentation of many tasks we have concluded that the following tasks improved our classification performances:

- Elimination of stop words: we have used a public predefined list sourced in github, that contains stop words of both MSA and Algerian dialects.
- Removing redundant letters: after the visualization of our data set we noticed that many words contain redundant letters either in Arabic or Latin characters such as: ‘allllllah’. Removing them improves our metrics performance.
- Handling special characters: according to this task we have decided to remove the punctuation marks and the flags since they don't typically carry sentiment themselves. In addition, we retain emojis due to their ability to enrich the text with expressive context, offering valuable insights into sentiment.
- Removing tags, email addresses, and urls: since our data set is mainly composed of social media platform comments, it often contains tags (e.g., <br>), email addresses, and urls. These elements, laden with random strings, are irrelevant to our classification task.

- Elimination of non-Arabic characters comments: since our dataset contains a small number of comments written in Latin charters we decided to remove them to improve our model's performances.
- Removing duplicated comments: by realizing this task we ensure the elimination of all redundant information that may skew our analysis results.
- The concluding step in data preprocessing involved partitioning the dataset, allocating 80% for model training, and reserving the remaining 20% for testing.

We should note that only for deep learning models (CNN and LSTM) all the preprocessing tasks were applied except the stop word removal.

2.3.3 Dataset encoding

In our machine learning models, the data encoding process is primarily handled by the tfidfvectorizer. Which transforms text documents into a matrix of TF-IDF (Term Frequency-Inverse Document Frequency) features, where each row represents a document, each column represents a unique word in the corpus, and each cell contains the TF-IDF score of the corresponding word in the document. TF-IDF reflects the importance of a word in a document relative to the entire corpus. By encoding the text data in this manner, the model can understand and learn from the textual information contained in the dataset.

The formula for calculating the TF-IDF score for a term t in a document d within a corpus is given by [48] as shown in equation (13) below:

$$TF\text{-}IDF\ (t,d,D) = TF\ (t,d) \times IDF\ (t,D) \qquad (13)$$

Where:

TF (t,d) is the term frequency of term t in document d. It measures how frequently a term occurs in a document.

IDF (t,D) is the inverse document frequency of term t in the corpus D. It measures how important a term is across the entire corpus. It is calculated as shown in equation (14) below:

$$IDF\ (t,D) = log(\frac{N}{df(t,D)}) \qquad (14)$$

Where N is the total number of documents in the corpus and df(t,D) is the number of documents containing the term t.

For deep learning models, a Word2Vec model is instantiated and trained on the tokenized comments. This word embedding technique was applied to transform words into dense vectors, adept at capturing semantic relationships between them in a continuous vector space. Following the training phase, our Word2Vec model boasts an extensive vocabulary size of 218,846 words. Subsequently, a secondary tokenization step is employed to convert the comments into sequences of integers, coupled with padding to ensure uniform length. An embedding matrix is constructed to map each word index to its respective word vector obtained from the Word2Vec model. This matrix serves as the foundation for initializing the embedding layer within a Sequential neural network model.

According to Bert's models, we have used the pre-trained word piece tokenizer either for distilbert or Arabic-Bert variants, while for the GPT model we employed the Byte Pair Encoding. Each tokenizer is adept at segmenting raw textual data into its constituent tokens while considering language-specific nuances, a pivotal step in the subsequent encoding process. Subsequently, the padding and truncation parameters are meticulously configured to ensure that all input sequences are uniformly sized, with a maximum length of 180 tokens. This uniformity facilitates streamlined processing within the neural network architecture.

2.3.4 Feature selection

After encoding the data, feature selection is performed to choose the most relevant features for the classification task. This process was applied only to machine learning models by selecting the top K features based on a specified scoring function. In this case, the chi-square (chi2) statistical test is employed to measure the independence between each feature and the target variable. By reducing the dimensionality of the feature space to the most informative features, feature selection aims to improve model performance, reduce computational complexity, and mitigate the risk of overfitting, ultimately enhancing the predictive accuracy of the machine learning model.

The formula for calculating the chi-squared statistic between a feature X and a target variable y is as shown in equation (15) below [49]:

$$X^2 = \sum \frac{(O-E)^2}{E} \tag{15}$$

Where:

O is the observed frequency, i.e., the actual number of occurrences of a particular combination of feature and target.

E is the expected frequency under the null hypothesis of independence between the feature and the target variable. It is calculated as expressed in equation (16) below:

$$X^2 = \sum \frac{(row\ total) \times (column\ total)}{grand\ total} \quad (16)$$

2.4 Conclusion

In conclusion, our multi-stage approach offers a thorough comparison of various sentiment analysis models tailored to the Algerian Arabic dialect. By distinguishing between positive and negative sentiments across both machine learning and deep learning models, we provide a detailed examination of each step in our methodology, including data collection, data encoding, and feature selection, ensuring a comprehensive understanding of the processes involved.

CHAPTER 3: Implementation

3.1 Introduction

In this chapter, we detail the tools employed for our sentiment analysis models implementation, spanning both machine learning and deep learning models, including the different frameworks and libraries, accompanied by illustrative source code snippets. Then, we describe the various models used in our approach including the specific parameters for each classifier, applied to a dataset of 89,118 Algerian dialect comments, evenly split between 44,559 positive and 44,559 negative sentiments.

3.2 Used resources

In our experiment, we used an HP computer equipped with a multi-core 11th generation Intel Core i5 processor, clocked at 2.60 GHz, and 16 GB of RAM. We trained our model in Google Collab using CPU for machine learning models and the GPU T4 for deep learning models, furthermore, we created an inference and interface in Jupyter Notebook to streamline the model evaluation process.

To develop our models we have used the Python environment which is a high-level programming language known for its simplicity, and utility in different paradigms including artificial intelligence, by providing various libraries that we will mention below [51].

- Numpy: an abbreviation of Numerical python, stands as a powerful numerical counting library that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently, it forms the foundation for numerous libraries in the Python ecosystem especially in data mining and machine learning domains [52].
- Pandas: is a flexible open-source Python library that provides high performance on both "relational" or "labeled" data, It is particularly well-suited for working with tabular data, such as data stored in CSV files by providing multiple functionalities for data manipulation such as leaning, transformation, aggregation, and visualization [53].
- Matplotlib: a ubiquitous library in Python, used for creating interactive and animated visualizations by providing multiple plotting functions to generate various types of plots, including line plots, scatter plots, bar plots, histograms, and more. Matplotlib is highly customizable allowing users to take control over multiple appearance aspects such as colors, labels, and axis formatting which makes it widely used in scientific computing, data analysis, and data visualization [54].

- Scikit-learn: is an open-source library widely employed in machine learning tasks, basically built on other Python libraries such as numpy, scipy, and Matplotlib. Scikit-learn provides numerous supervised and unsupervised algorithms including classification, regression, and model selection. It also provides tools for model evaluation, validation, and hyperparameter tuning. This makes her widely used by researchers, and data scientists for building their machine-learning models [55].
- tensorflow: represents a machine learning framework developed by Google capable of building and training neural networks, it provides numerous tools, libraries, and resources to promote the development of machine learning models across multiple platforms. Tensorflow is particularly known for its simplicity, flexibility, and scalability, making it a popular choice among researchers in the artificial intelligence field [56].
- Keras: Python neural network library, that provides a high-level API for building and training deep learning models, it serves the users' simple interface that handles the complexity of working directly with lower-level libraries like tensorflow [57].
- Pytorch: an open-source machine learning framework developed by Facebook, it offers a dynamic approach for building and training neural networks across a Python-based interface that enables rapid prototyping and experimentation. Pytorch consists of dynamic computation graphs, automatic differentiation, and a rich ecosystem of libraries and tools for various machine-learning tasks. Making it popular among researchers in the field of artificial intelligence [58].
- re: a Python module that provides support for working with regular expressions (re), it's a powerful tool for pattern matching and text manipulation making it indispensable for tasks such as data validation, text parsing, and information extraction [59].
- NLTK: or the Natural Language Toolkit, which represents a popular platform for building Python programs to work with human language data, providing a simple interface that covers over 50 corpora and lexical resources accompanied with multiple text processing libraries for classification tokenization, tagging, parsing, and stemming [60].
- pyarabic: a Python library especially customized for Arabic text processing and analysis across various functionalities such as normalization, tokenization, stemming, and other language-specific tasks [61].

- Os (Operating system): it’s a Python module that provides multiple functionalities to manipulate files including directory management, file read-write operations, and accessing the underlying operating system's environment variables [62].
- JSON (Javascript Object Notation): represents a Python library that provides multiple functions for encoding and decoding JSON data, the JASON library allows Python to work flexibly with JSON data offering multiple functionalities such as converting Python data structures into JSON strings for transmission or storage, parsing JSON strings into Python data structures for further processing [63].
- Transformers: it’s an open-source library developed by Huggin Face that provides multiple pre-trained models for natural language understanding and natural language generation tasks, especially those built on transformers architecture including variants such as BERT, distilbert, Gpt, and roberta. The main goal for this library is to facilitate the fine-tuning of these available pre-trained models on custom datasets.

Additionally, it offers multiple functionalities for tokenization, model configuration, and model evaluation, making it a comprehensive tool for NLP research and development in Python [64].

- Tkinter (Tk): is the standard GUI (Graphical User Interface) library for Python. It provides a fast and easy way to create GUI applications. Tkinter is included with the standard Python distribution and offers a simple way to create windows, dialogs, buttons, menus, and various other GUI components. It is a wrapper around the Tk GUI toolkit, which is originally from the Tcl programming language. Tkinter is widely used for creating desktop applications in Python due to its simplicity and ease of use [65].
- Jubyter Notebook: is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It is widely used in data science, scientific computing, and machine learning for its interactive environment which facilitates data analysis, visualization, and exploration. The name "Jupyter" comes from the core programming languages it supports: Julia, Python, and R, but it also supports many other languages [66].

3.3 Experiments

3.3.1 Models Construction

In our pursuit of robust sentiment analysis for the Algerian dialect, we have constructed machine-learning and deep-learning models.

3.3.1.1 Machine Learning Models

Following the features extraction from comments using TF-IDF. And optimal feature selection to enhance performance metrics. The ensuing experimental phase introduces various classification models, including K-nearest neighbors (with K set to 20), a Support Vector Machine with a linear kernel, and Naïve Bayes multinomial classifiers.

Models	Parameters
KNN	Max features = 8000 Feature extraction = tfidfvectorizer Feature selection algorithm = chi2 K (select best features) = 1000 K (neighbors number) = 20 Data set size = (44559 positive and 44559 negative) Test size = 20% Random state = 42 Metrics = 'accuracy, precision, recall, f1 score'
SVM	Max features = 8000 Feature extraction = tfidfvectorizer Feature selection algorithm = chi2 K (select best features) = 1000 Kernel = Linear Data set size = (44559 positive and 44559 negative) Test size = 20% Random state = 42 Metrics = 'accuracy, precision, recall, f1 score'
Naïve Bayes Multinomial	Max features = 8000 Feature extraction = tfidfvectorizer Feature selection algorithm = chi2

	Data set size = (44559 positive and 44559 negative) Test size = 20% Random state = 42 Metrics = ‘accuracy, precision, recall, f1 score’

Table *3.1* – Parameters of the employed machine learning models.

3.3.1.2 Deep Learning Models

According to the LSTM model, after completing the word2vec model training on all corpora collected, we have initialed our model with an embedding layer employing this word2vec representation to tokenized comments, furthermore, Two LSTM layers are added one after the other to capture long-term dependencies in sequential data. Each LSTM layer comprises 128 units (or cells) with a 0.2 dropout to prevent overfitting. Finally, a Dense layer with a single unit and a sigmoid activation function is added. The output will be a probability between 0 and 1, where values closer to 1 represent a positive sentiment and values closer to 0 represent a negative sentiment.

As for our CNN model, we initialized our architecture with a word embedding layer, leveraging a vocabulary size of 218846 and setting a maximum sequence length of 180 tokens same as the LSTM model. Following this, we added two conventional layers with 64 filters each, employing a kernel size of 3 and Rectified Linear Unit (ReLU) activation functions to capture essential features from the text data. Maxpooling layers with a pool size of 2 were inserted to downsample the feature maps. To mitigate overfitting, a dropout rate of 0.4 was applied after each maxpooling layer. The final layer utilized a sigmoid activation function to generate sentiment probabilities.

After an experimental hyperparameter process, We should note that the Adam optimizer with a learning rate set at 0.001 and a batch size of 32 yields optimal results for both the LSTM and CNN deep learning models spanning seven epochs of training.

Models	Layers (Type)	Parameters

CNN	Sequential model with 1) Conv1D Layer. 2) Dropout Layer. 3) maxpooling1d Layer. 4) Conv1D Layer. 5) Dropout Layer. 6) globalmaxpooling1d Layer 7) Dense Classification Layer (output layer).	Vocabulary size = 218846 Data set size = (44725 positive and 44725 negative) Max length = 180 Conventional layers (filters=64, kernel_size=3, activation function=Relu) Maxpooling layers (pool size = 2) Dropout rate=0.4 Activation function = 'sigmoid' Optimizer = 'Adam' Learning Rate = 0.001 Batch size = 32 Loss Function = 'binary_crossentropy' Epochs number = 7 Metrics = 'accuracy, precision, recall, f1 score'
LSTM	Sequential model with 1)Embedding Layer. 2)LSTM Layer (with return-sequences=True). 3)Dropout Layer. 4)LSTM Layer. 5)Dropout Layer. 6)Dense Classification Layer (output layer).	Vocabulary size = 218846 Data set size = (44725 positive and 44725 negative) Max length = 180 LSTM Units=128 Dropout rate=0.2 Activation function = 'sigmoid' Optimizer = 'Adam' Learning Rate = 0.001 Batch size = 32 Loss Function = 'binary_crossentropy' Epochs number = 7 Metrics = 'accuracy, precision, recall, f1 score'

Table *3.2* – Deep Learning Models Architecture and Parameters.

3.3.1.3 Transformers Models

On the other side, we have explored fine-tuning many Bert models trained already on a huge amount of MSA Arabic with different architectures such as Arabic-Bert compilation as shown in the table below.

Since then, distilbert models, known for their streamlined architecture, have enabled faster training times. We have further fine-tuned a variant called distilbert-multilingual, which has been trained on over 104 languages, including Arabic, Spanish, English, and French. This model features 6 layers, 768 embedding dimensions, and 12 attention heads.

We also fine-tuned the aragpt2 base Model, which is based on the decoder architecture and specializes in text generating.
During experimentation and hyperparameter fine-tuning, we get this combination that achieves the best results for all our employed Bert models:

- Learning rate: 10^{-5}.
- Batch size: 32.
- Sequence length: 180.
- Epochs number: 2.

	BERT-Mini	BERT-Medium	BERT-Base	Distilbert multi cased	Aragpt2-base
Hidden layers	4	8	12	6	12
Attention heads	4	8	12	12	12
Hidden size	256	512	768	768	768
Parameters	11M	42M	110M	134M	135M

Table *3.3* – BERT Arabic variants, distilbert multilanguage cased, and aragpt2-base architecture.

3.4 Resutls and discusion

We have evaluated our models employing four metrics performances, given by the equations (17 – 20) below [50].

- Precision: represents a measure to check the quality of the model’s positive predictions It's calculated by looking at the ratio of the correctly predicted positive cases to all the cases the model predicted as positive.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (17)$$

- Recall: is the proportion of true positive elements identified by the model, out of all the actual positive elements in the dataset. It is calculated by dividing the number of true positives by the sum of true positives and false negatives.

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (18)$$

- Accuracy: is a metric that measures the overall correctness of a model's predictions by comparing the total number of correct predictions to the total number of predictions made. In simpler terms, accuracy tells us how often the model gets its predictions right across all classes or outcomes.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad (19)$$

- F1-score: is a metric that combines both precision and recall into a single value, providing a balance between the two. It's calculated by taking the harmonic mean of precision and recall, giving equal weight to both. This makes it a useful measure for evaluating the overall performance of a classification model, especially when the class distribution is uneven or when both false positives and false negatives are important considerations.

$$\text{F1 score} = 2\,.\left(\frac{Precision\,.\,Recall}{Precision+Recall}\right) \qquad (20)$$

	Accuracy	Precision	Recall	F1-score
KNN	77%	78%	76%	76%
SVM linear	**82%**	**82%**	**81%**	**81%**
Naïve Bayes multinomial	80%	80%	79%	79%

Table *3.4* – Machine learning-based models results.

Based on the results shown in Table 3.4, we can conclude that the SVM linear model demonstrates the highest overall performance among the three models evaluated (KNN, SVM linear, and NB multinomial). It achieves the best scores across all metrics: accuracy (82%), precision (82%), recall (81%), and F1-score (81%). The multinomial Naive Bayes model shows comparable performance, slightly trailing the SVM linear model with similar scores across the board. The KNN model has the lowest performance metrics, with accuracy at 77%, precision at 78%, recall at 76%, and F1-score at 76%. Therefore, the SVM linear model is the most effective among the tested models for this particular task.

	Accuracy	Precision	Recall	F1-score	Training time
LSTM	**85%**	**88%**	**81%**	**83%**	**5 min**
CNN	82%	86%	76%	80%	1 min

Table *3.5* – Deep Learning -based models results Along With training time.

Based on Table 3.5, the LSTM model outperforms the CNN model in terms of accuracy, precision, recall, and F1 score. The LSTM achieves an accuracy of 85%, a precision of 88%, a recall of 81%, and an F1-score of 83%, whereas the CNN achieves an accuracy of 82%, a precision of 86%, a recall of 76%, and an F1-score of 80%. However, the CNN model benefits from a significantly faster training time of 1 minute compared to the LSTM's 5 minutes. Therefore, the choice between these models should be guided by the specific needs of the application. For scenarios where model performance is paramount, the LSTM is the better choice. On the other hand, if rapid training is a priority, CNN is more suitable despite its slightly lower performance metrics.

	Accuracy	Precision	Recall	F1-score	Training time
Bert-mini	86%	86%	86%	86%	5 min 26 s
Bert-medium	86,7%	86,7%	86,6%	86,6%	24 min 52 s
Bert-base	**87,9%**	**87,9%**	**87,9%**	**87,9%**	**1 h 18 min 15 s**
Distilbert	85,9%	85,9%	85,9%	85,9%	41 min 14 s
Aragpt2-base	85,7%	85,7%	85,7%	85,7%	1 h 25 min 6 s

Table 3.6 – Transformers-Based Models Results Along With Training Time.

According to Table 3.6, the Bert-base model exhibits the highest performance in terms of accuracy, precision, recall, and F1-score, with all metrics at 87.9%. This is followed by Bert-medium with metrics around 86.7%, and Bert-mini with metrics at 86%. Distilbert and aragpt2-base show slightly lower performance, with all metrics at 85.9% and 85.7% respectively. However, this improved performance comes at the cost of significantly longer training times. Bert-base requires 1 hour 18 minutes 15 seconds to train, making it the most time-consuming model. In contrast, the Bert-mini, the quickest to train, requires only 5 minutes 26 seconds. Therefore, the choice of model should be based on the trade-off between performance and training time. For the highest accuracy, precision, recall, and F1-score, Bert-base is the best option despite its long training time. For faster training with reasonably high performance, Bert-mini or Distilbert would be more suitable. Bert-medium offers a middle ground with slightly better performance than Bert-mini but with a longer training time.

This statement indicates that among the various models tested, BERT variants consistently achieved the highest accuracies. As a result, the decision was made to further improve the accuracy by employing ensemble learning techniques such as stacking and voting specifically on these BERT models. Stacking involves combining predictions from multiple models, in this case, Aragpt2, Arabicbert Base, and Distilbert, using a meta-model to generate a final prediction. Voting, on the other hand, combines predictions from multiple models by either taking the majority vote (hard voting) or averaging predicted probabilities (soft voting).

	Accuracy	Precision	Recall	F1-score
Majority voting	90%	89,1%	92,5%	90%
Stacking	91,1%	90,4%	91,9%	91,2%

Table *3.7* – Ensemble learning results.

According to Table 3.7, applying ensemble learning approaches on the three transformer models Aragpt2, Arabicbert Base, and Distilbert demonstrates significant performance improvement. The majority voting ensemble achieved an accuracy of 90%, precision of 89.1%, recall of 92.5%, and F1-score of 90%. The stacking ensemble further enhanced the results, achieving an accuracy of 91.1%, precision of 90.4%, recall of 91.9%, and F1-score of 91.2%. These findings indicate that both ensemble methods improve the overall performance of individual transformer models, with stacking providing the best results across all metrics.

Figures 3.1, and 3.2 show the learning curves for the LSTM model and CNN model respectively.

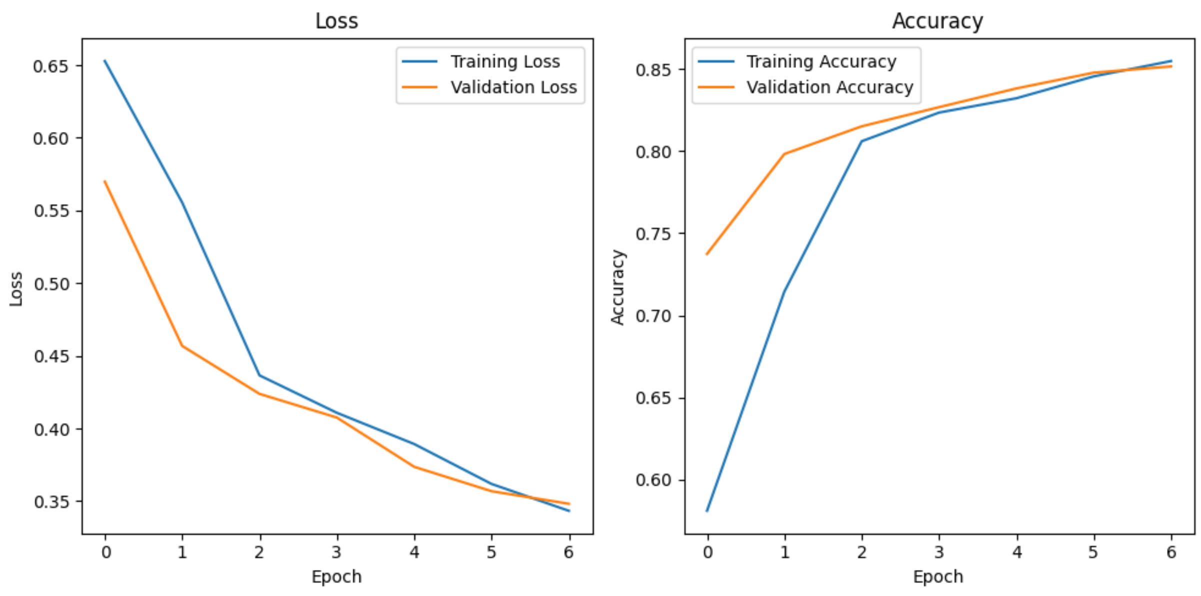


Figure 3.*1* – Learning curves for the LSTM model.

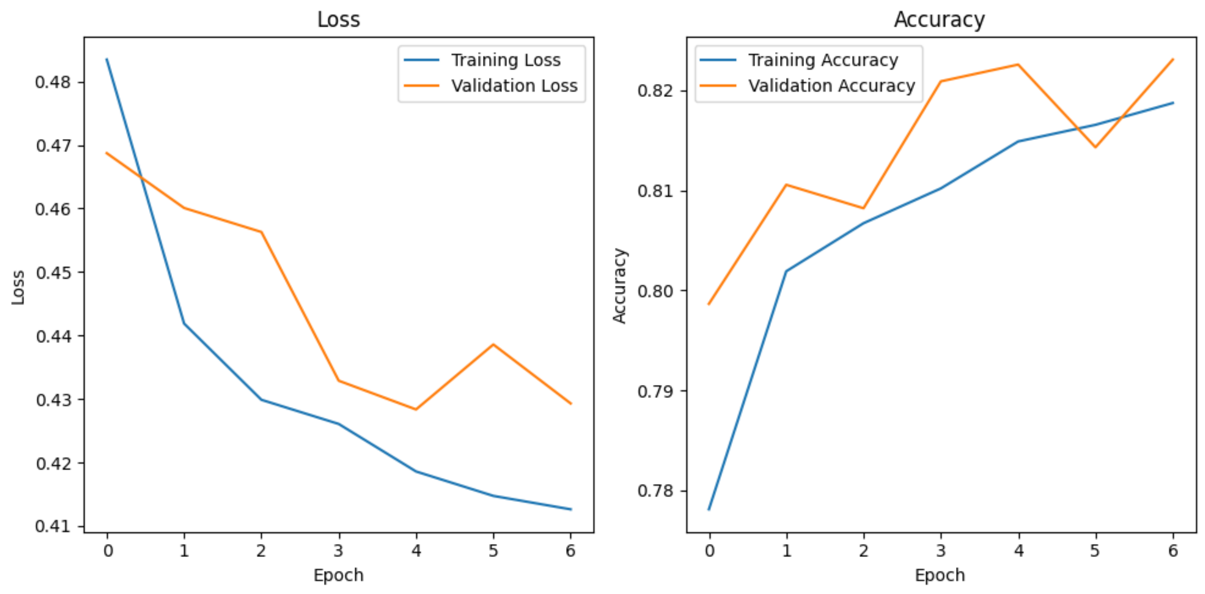


Figure 3.2 – Learning curves for the CNN model.

According to the learning curves for both LSTM and CNN models, presented in Figures 3.1, and 3.2, respectively. There is a clear trend of gradual improvement in accuracy throughout the training process. This indicates that the models are effectively learning and generalizing patterns, thereby enhancing their performance over time. Additionally, the consistent decrease in validation loss across epochs suggests that the models are converging well.

	Accuracy	Precision	Recall	F1-score
Svm linear	66.1%	91.7%	38.1%	53.9%
Knn	66.8%	86.5%	42.7%	57.1%
Naïve Bayes	72%	73.9%	71.3%	72.6%
Lstm	46%	46.2%	46%	45.7%
Cnn	45.3%	45.5%	45.3%	44.5%
Bert-Arabic mini	78.8%	79.8%	78.8%	78.8%
Bert-Arabic medium	77.7%	78.3%	77.7%	77.7%
Bert-Arabic base	78.6%	79%	78.6%	78.6%
Distilbert multilanguage	78.3%	78.9%	78.3%	78.3%
Aragpt-2	78.6%	78.8%	78.6%	78.6%
Ensemble learning (majority voting)	81,2%	87%	75%	80,5%
Ensemble learning (stacking)	81,2%	87%	75%	80,5%

Table 3.8 – Metrics performance results obtained on the Narabizi datasets.

	Accuracy	Precision	Recall	F1-score
Svm linear	62.2%	88.9%	36%	51.2%
Knn	63.6%	84.5%	41.7%	55.8%
Naïve Bayes	71.8%	71.8%	80.6%	75.9%
Lstm	51.2%	52.6%	51.2	51%
Cnn	49.1%	51.2%	49.1%	48.1%
Bert-Arabic mini	79.7%	80%	79.7%	79.7%
Bert-Arabic medium	82.9%	82.9%	82.9%	82.9%
Bert-Arabic base	87.8%	87.9%	87.8%	87.8%
Distilbert multilanguage	80.9%	80.9%	80.9%	80.9%
Aragpt-2	78.9%	79%	78.9%	78.9%
Ensemble learning (majority voting)	85,4%	86,6%	86,8%	86,7%
Ensemble learning (stacking)	87,8%	87%	91,5%	89,2%

Table *3.9* – Metrics performance results obtained on the Twifil datasets.

Based on the results in Tables 3.8 and 3.9, several observations can be made from testing the sentiment analysis models on the Narabizi and TWIFIL datasets.

- Firstly, it's evident that the performance of the models varies across different datasets, indicating the influence of dataset characteristics on model effectiveness.
- In the Narabizi dataset, the Bert-Arabic mini model achieves the highest performance across all metrics, including accuracy (78.8%), precision (79.8%), recall (78.8%), and F1-score (78.8%). This indicates that transformer-based models, particularly BERT, are highly effective in handling the linguistic characteristics of the Narabizi dataset. Other BERT models, including Bert-Arabic base (78.6% accuracy) and distilbert multilingual (78.3% accuracy), also perform very well, underscoring the robustness of transformer architectures.
- Conversely traditional machine learning models like SVM, KNN, and Naïve Bayes exhibit lower performance metrics on the Narabizi dataset. Although Naïve Bayes performs reasonably well with an accuracy of 72%, it still falls short of the BERT and GPT models.
- In the TWIFIL dataset, the Bert-Arabic base model achieves the highest performance, with an accuracy of 87.8%, precision of 87.9%, recall of 87.8%, and F1-score of 87.8%.

This reinforces the efficacy of transformer-based models for sentiment analysis in the Algerian dialect. The Bert-Arabic medium (82.9% accuracy) and distilbert multilingual (80.9% accuracy) also perform very well, highlighting the general effectiveness of BERT-based models across different datasets. The aragpt-2 model also shows strong performance, particularly in the TWIFIL dataset, with an accuracy of 78.9%, indicating its potential in handling Algerian dialect sentiment analysis Although it is intended for text editing.

- Traditional machine learning models on the TWIFIL dataset perform moderately well, with Naïve Bayes achieving an accuracy of 71.8% and KNN reaching 63.6%. However, these models are again outperformed by the BERT variants and aragpt-2, emphasizing the superior capability of transformer models in capturing complex language nuances.
- On the other hand, deep learning models such as LSTM and CNN show significantly lower performance on both datasets, which can be attributed to several factors. Firstly, there might be a lack of generalization due to overfitting to the training data, meaning the models perform well on familiar data but struggle with new, unseen data. Secondly, the linguistic characteristics and distribution of sentiments in the training and testing datasets may differ significantly, making it challenging for the models to adapt. Furthermore, the word2vec model relies on the context within the training data to generate word embeddings, which may not transfer well to datasets with different contexts and vocabulary. These issues collectively contribute to the lower performance observed when the models are tested on new datasets like TWIFIL and Narabizi.
- However, it is important to note that while transformer models like BERT and aragpt-2 offer superior performance, they also come with more complex architectures, making them more time-consuming and computationally intensive compared to traditional machine learning models. This complexity can be a significant factor in practical applications, where computational resources and processing time are critical considerations.
- Moreover, the observations from the ensemble learning models applied to sentiment analysis on the Narabizi and Twifil datasets demonstrate a clear improvement over individual models. On the Narabizi dataset, both majority voting and stacking achieved the highest accuracy of 81.2%, with precision, recall, and F1-score all significantly improving compared to individual models. On the Twifil dataset, stacking achieved the best overall performance with an accuracy of 87.8%, precision of 87%, recall of 91.5%,

and F1-score of 89.2%, matching the accuracy of the BERT-Arabic base model but surpassing it in recall and F1-score. These results indicate that ensemble learning, particularly stacking, effectively leverages the strengths of individual transformer models to enhance performance across different metrics and datasets.

- Overall, the results indicate that transformer-based models, particularly the various BERT models, and aragpt-2, are best suited for sentiment analysis in both the Narabizi and TWIFIL datasets. Their ability to understand and process intricate linguistic features makes them the preferred choice for sentiment analysis tasks involving the Algerian dialect, despite their higher computational demands, Moreover, the insights gleaned from the ensemble learning methods exhibit a tangible enhancement over individual models. Notably, stacking emerges as a potent strategy, effectively harnessing the capabilities of individual transformer models to bolster performance across a spectrum of metrics and datasets.

Model	Accuracy	Precision	Recall	F1 score	Realized by
Majority voting	**90%**	**89,1%**	**92,5%**	**90%**	**Our model**
Stacking	**91,1%**	**90,4%**	**91,9%**	**91,2%**	**Our model**
SVM	**82%**	**82%**	**81%**	**81%**	**Our model**
LSTM	**85%**	**88%**	**81%**	**83%**	**Our model**
Bert Arabic Base	**87,9%**	**87,9%**	**87,9%**	**87,9%**	**Our model**
Svm	86%	89%	82%	85%	Abdeli [39]
Lstm	81%	79%	75%	77%	Abdeli [39]
Bert Arabic large	78,38%	/	/	81,74%	Benmounah [37]
Svm	83,7%	/	/	/	Mazari [38]

Table *3.10* – Performance of Previous Works on Algerian Dialect Sentiment Analysis.

According to Table 3.10, several observations can be made from the comparison between sentiment analysis models for the Algerian dialect.

- Abdeli's SVM model outperforms their LSTM model, achieving an accuracy of 86% and an F1 score of 0.85, compared to 81% accuracy and an F1 score of 0.77 for LSTM. In contrast, our models, utilizing SVM linear and LSTM algorithms, display similar overall performance. Our LSTM model exhibits slightly higher accuracy (85% vs. 82%) and precision (88% vs. 82%) compared to the SVM linear model, with both demonstrating comparable recall and F1-score metrics. Notably, our dataset encompasses a larger volume of comments, potentially influencing model performance.

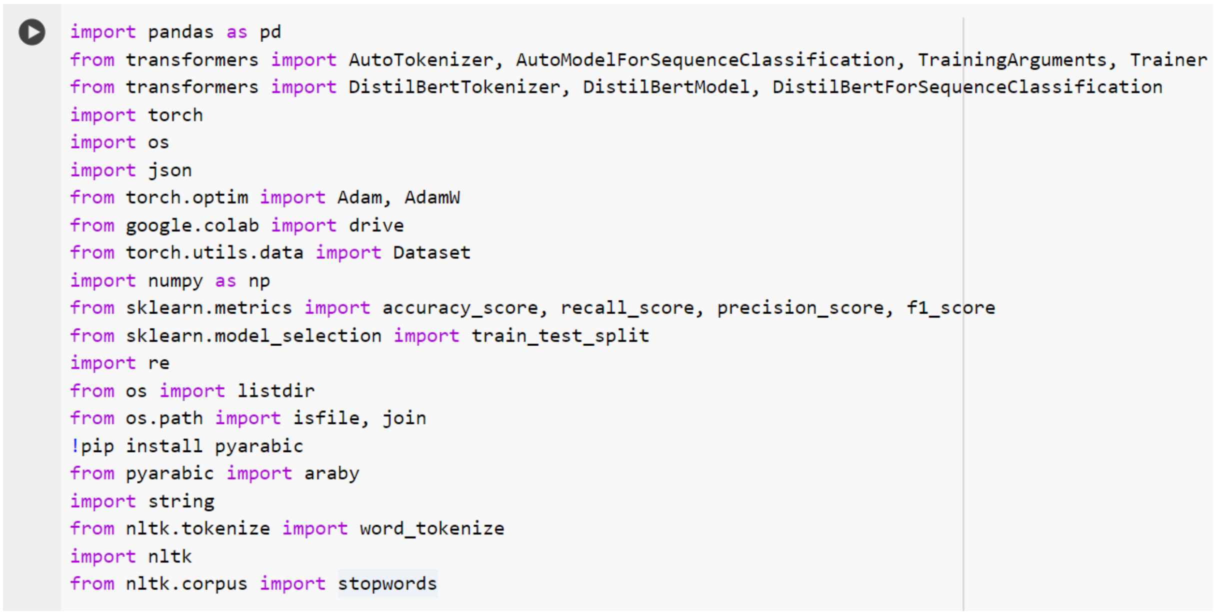
- Comparing our binary sentiment analysis model for the Algerian dialect, written in Arabic, to Mazari's study reveals notable findings. Our model, incorporating SVM linear, LSTM, and CNN algorithms, demonstrates strong performance metrics such as an accuracy of 85% and a precision of 88% for the LSTM model. In contrast, Mazari's approach, which involves transcribing the dataset into Latin characters and enhancing it with word2vec models, achieves competitive results, particularly with SVM achieving an accuracy of 83.70%. However, our CNN model showcases impressive precision (86%) and F1-score (80%), indicating its efficacy in sentiment analysis.

- When comparing our specialized binary sentiment analysis model for the Algerian dialect in Arabic characters to Benmouneh's model designed for three-label sentiment analysis across both Arabic and Latin characters, clear distinctions emerge. Our model, leveraging BERT-base, achieves exceptional performance metrics, boasting an accuracy, precision, recall, and F1-score all at 87.9%, alongside an efficient training time of just 1 hour and 18 minutes. In contrast, Benmouneh's model, utilizing BERT Arabic large, exhibits slightly lower performance metrics, with an F1-score of 0.7838 and an accuracy of 0.8174, albeit requiring a longer training time of 1 hour and 53 minutes. While both models harness the power of BERT architecture, Benmouneh's broader task of three-label sentiment analysis, covering both Arabic and Latin characters, presents a comprehensive approach albeit with marginally reduced performance metrics.

3.5 Examples of source codes

This section will present a few source code examples and result snippets of our experiment.

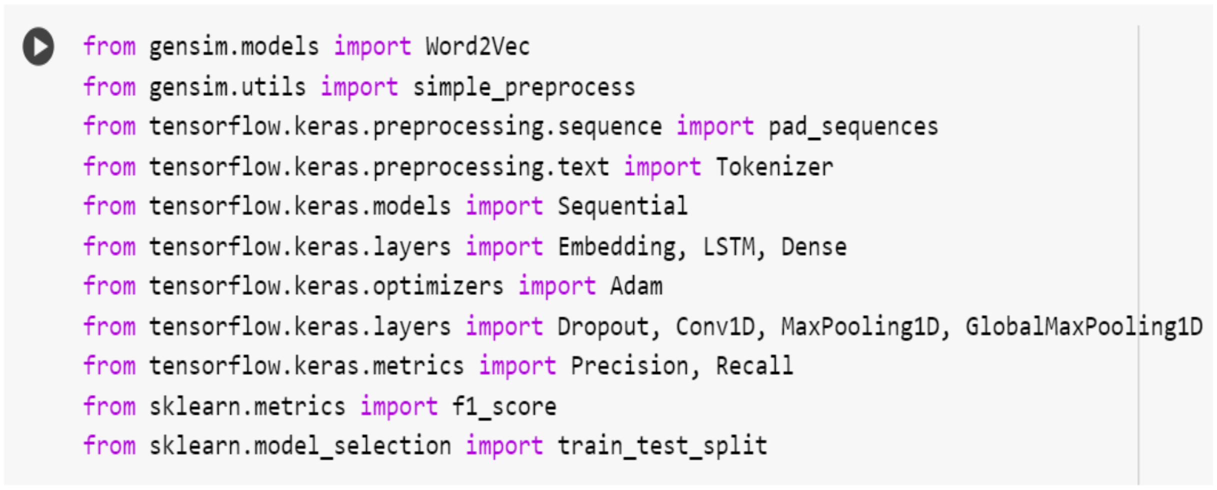
In Figure 3.3 we present how we import essential libraries for Bert models.



```python
import pandas as pd
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TrainingArguments, Trainer
from transformers import DistilBertTokenizer, DistilBertModel, DistilBertForSequenceClassification
import torch
import os
import json
from torch.optim import Adam, AdamW
from google.colab import drive
from torch.utils.data import Dataset
import numpy as np
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score
from sklearn.model_selection import train_test_split
import re
from os import listdir
from os.path import isfile, join
!pip install pyarabic
from pyarabic import araby
import string
from nltk.tokenize import word_tokenize
import nltk
from nltk.corpus import stopwords
```

***Figure* 3.3** *– Import Essential Libraries For Bert Models.*

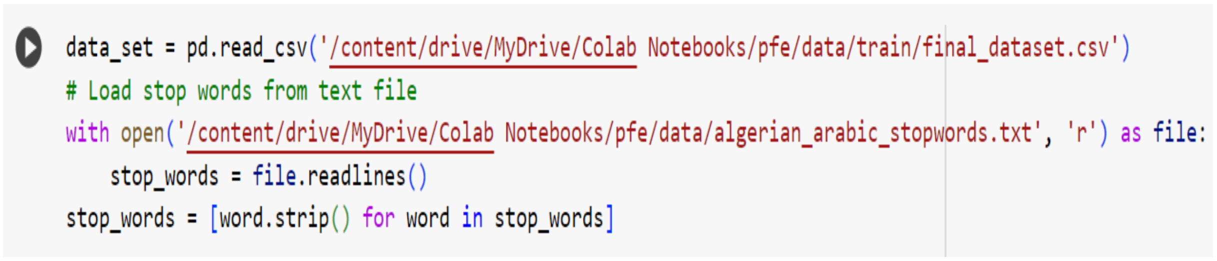
In Figure 3.4, we demonstrate the importation of key libraries needed for LSTM and CNN models.



```python
from gensim.models import Word2Vec
from gensim.utils import simple_preprocess
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Dropout, Conv1D, MaxPooling1D, GlobalMaxPooling1D
from tensorflow.keras.metrics import Precision, Recall
from sklearn.metrics import f1_score
from sklearn.model_selection import train_test_split
```

Figure 3.4 – Import Essential Libraries for LSTM And CNN Models.

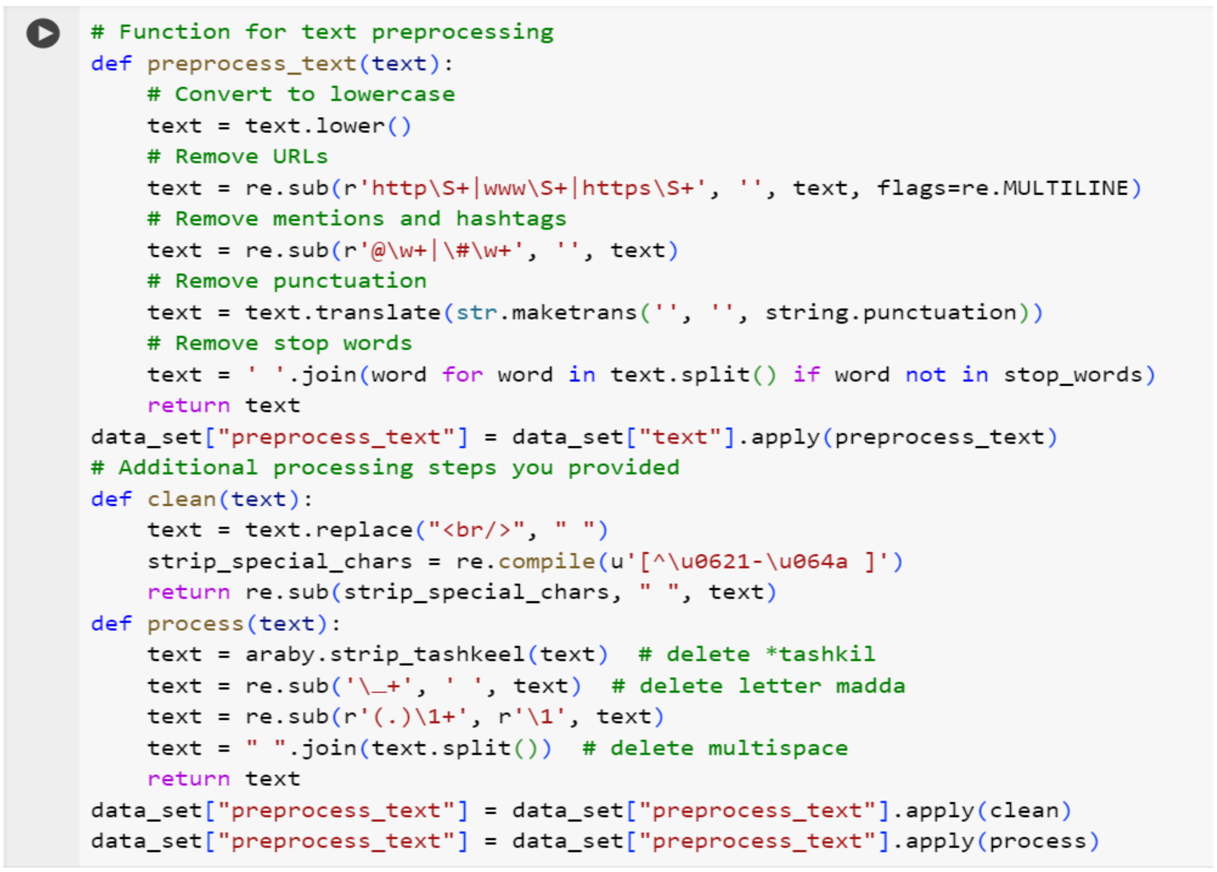
In Figure 3.5, we illustrate the process of reading the dataset and the stop word list.



```
data_set = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/pfe/data/train/final_dataset.csv')
# Load stop words from text file
with open('/content/drive/MyDrive/Colab Notebooks/pfe/data/algerian_arabic_stopwords.txt', 'r') as file:
    stop_words = file.readlines()
stop_words = [word.strip() for word in stop_words]
```

Figure 3.5 – Read Dataset And Stop Word List.

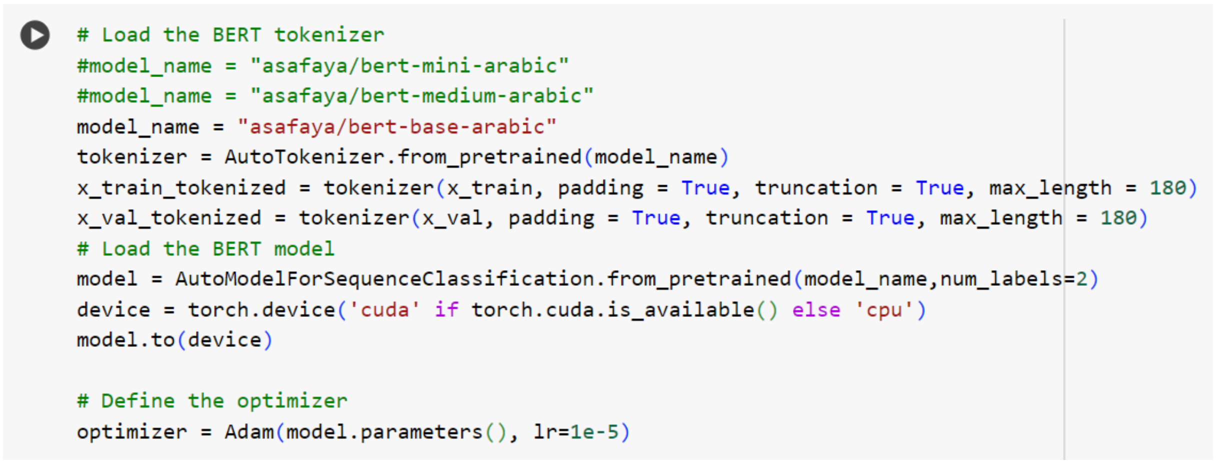
In Figure 3.6, we present the necessary preprocessing functions used to clean our dataset.



```
# Function for text preprocessing
def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()
    # Remove URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)
    # Remove mentions and hashtags
    text = re.sub(r'@\w+|\#\w+', '', text)
    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation))
    # Remove stop words
    text = ' '.join(word for word in text.split() if word not in stop_words)
    return text
data_set["preprocess_text"] = data_set["text"].apply(preprocess_text)
# Additional processing steps you provided
def clean(text):
    text = text.replace("<br/>", " ")
    strip_special_chars = re.compile(u'[^\u0621-\u064a ]')
    return re.sub(strip_special_chars, " ", text)
def process(text):
    text = araby.strip_tashkeel(text)  # delete *tashkil
    text = re.sub('\_+', ' ', text)  # delete letter madda
    text = re.sub(r'(.)\1+', r'\1', text)
    text = " ".join(text.split())  # delete multispace
    return text
data_set["preprocess_text"] = data_set["preprocess_text"].apply(clean)
data_set["preprocess_text"] = data_set["preprocess_text"].apply(process)
```

Figure 3.6 – Define Preprocessing Functions.

In Figure 3.7, we demonstrate how we load our BERT tokenizer and model for subsequent fine-tuning.

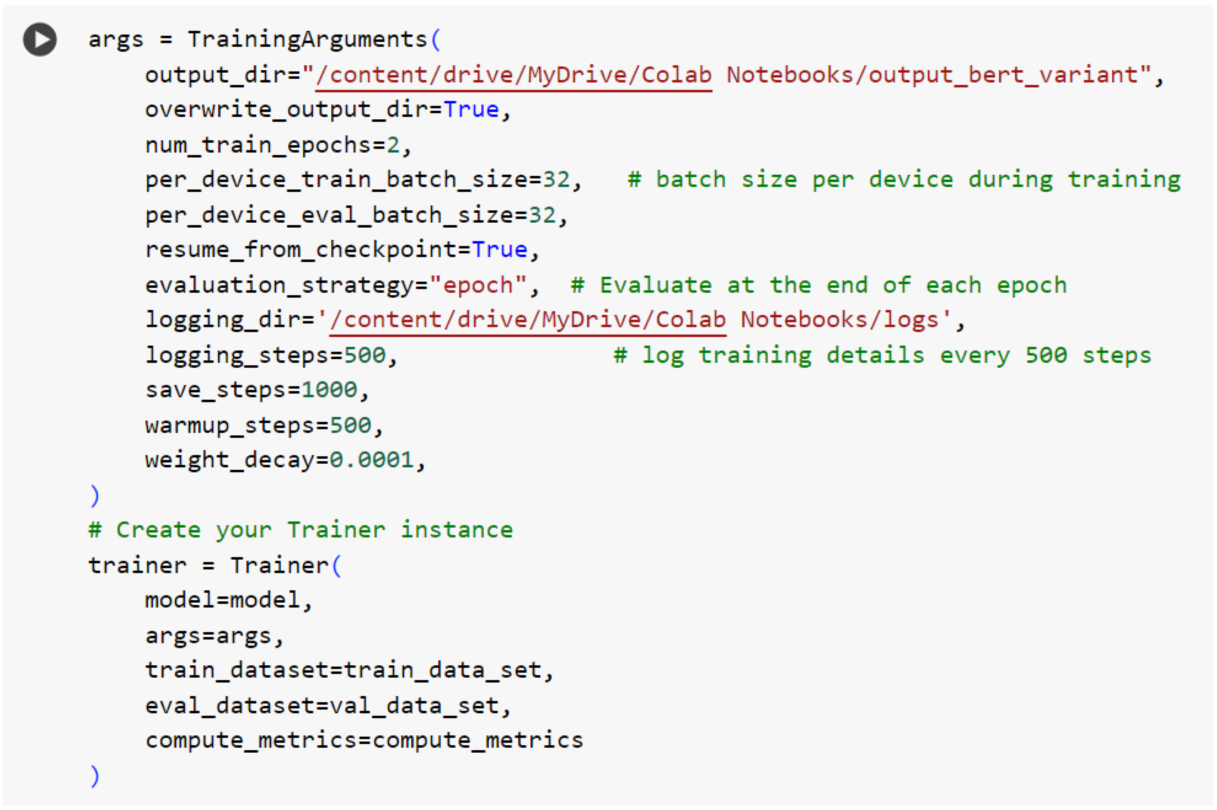


```
# Load the BERT tokenizer
#model_name = "asafaya/bert-mini-arabic"
#model_name = "asafaya/bert-medium-arabic"
model_name = "asafaya/bert-base-arabic"
tokenizer = AutoTokenizer.from_pretrained(model_name)
x_train_tokenized = tokenizer(x_train, padding = True, truncation = True, max_length = 180)
x_val_tokenized = tokenizer(x_val, padding = True, truncation = True, max_length = 180)
# Load the BERT model
model = AutoModelForSequenceClassification.from_pretrained(model_name,num_labels=2)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)

# Define the optimizer
optimizer = Adam(model.parameters(), lr=1e-5)
```

Figure 3.7 – The Bert model and tokenizer loading.

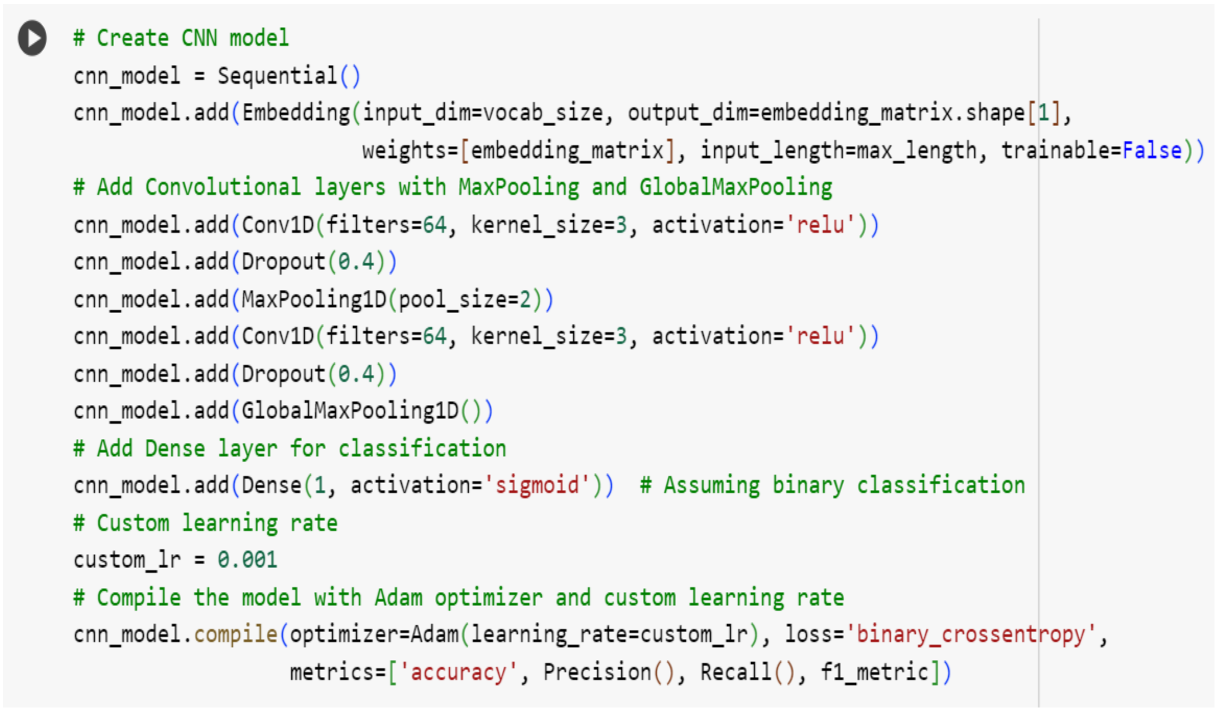
In Figure 3.8, we present how we configure the training parameters for our Bert variant model.



```
args = TrainingArguments(
    output_dir="/content/drive/MyDrive/Colab Notebooks/output_bert_variant",
    overwrite_output_dir=True,
    num_train_epochs=2,
    per_device_train_batch_size=32,   # batch size per device during training
    per_device_eval_batch_size=32,
    resume_from_checkpoint=True,
    evaluation_strategy="epoch",  # Evaluate at the end of each epoch
    logging_dir='/content/drive/MyDrive/Colab Notebooks/logs',
    logging_steps=500,                # log training details every 500 steps
    save_steps=1000,
    warmup_steps=500,
    weight_decay=0.0001,
)
# Create your Trainer instance
trainer = Trainer(
    model=model,
    args=args,
    train_dataset=train_data_set,
    eval_dataset=val_data_set,
    compute_metrics=compute_metrics
)
```

Figure 3.8 – Configuring Training Parameters for BERT Variant Model.

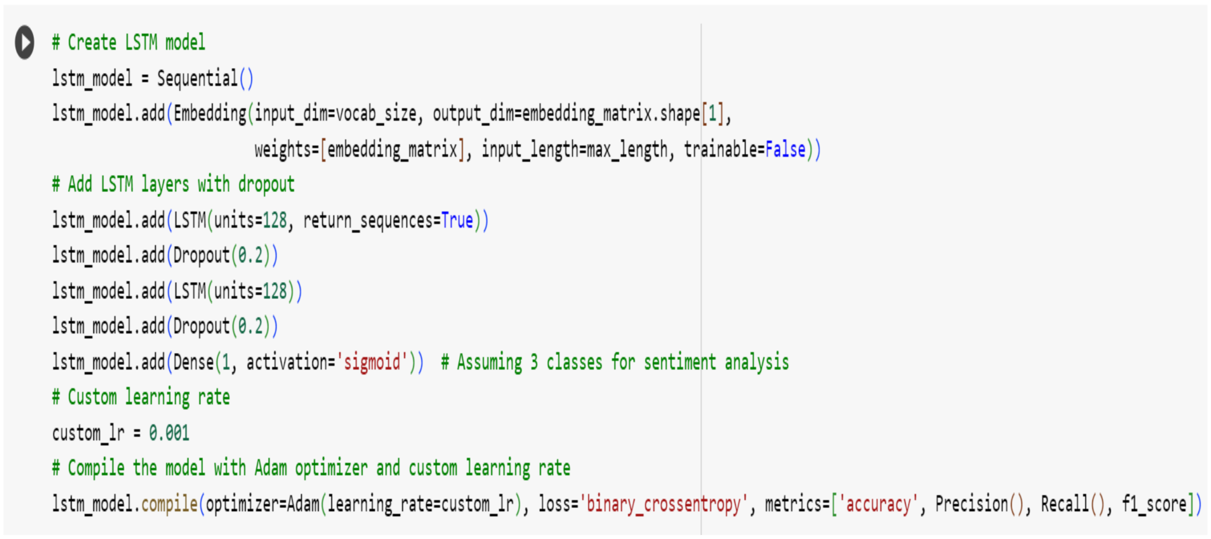
In Figure 3.9, we illustrate the detailed configuration of our CNN model architecture.



```python
# Create CNN model
cnn_model = Sequential()
cnn_model.add(Embedding(input_dim=vocab_size, output_dim=embedding_matrix.shape[1],
                        weights=[embedding_matrix], input_length=max_length, trainable=False))
# Add Convolutional layers with MaxPooling and GlobalMaxPooling
cnn_model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(Dropout(0.4))
cnn_model.add(MaxPooling1D(pool_size=2))
cnn_model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(Dropout(0.4))
cnn_model.add(GlobalMaxPooling1D())
# Add Dense layer for classification
cnn_model.add(Dense(1, activation='sigmoid'))  # Assuming binary classification
# Custom learning rate
custom_lr = 0.001
# Compile the model with Adam optimizer and custom learning rate
cnn_model.compile(optimizer=Adam(learning_rate=custom_lr), loss='binary_crossentropy',
                  metrics=['accuracy', Precision(), Recall(), f1_metric])
```

Figure 3.9 – CNN Model Architecture.

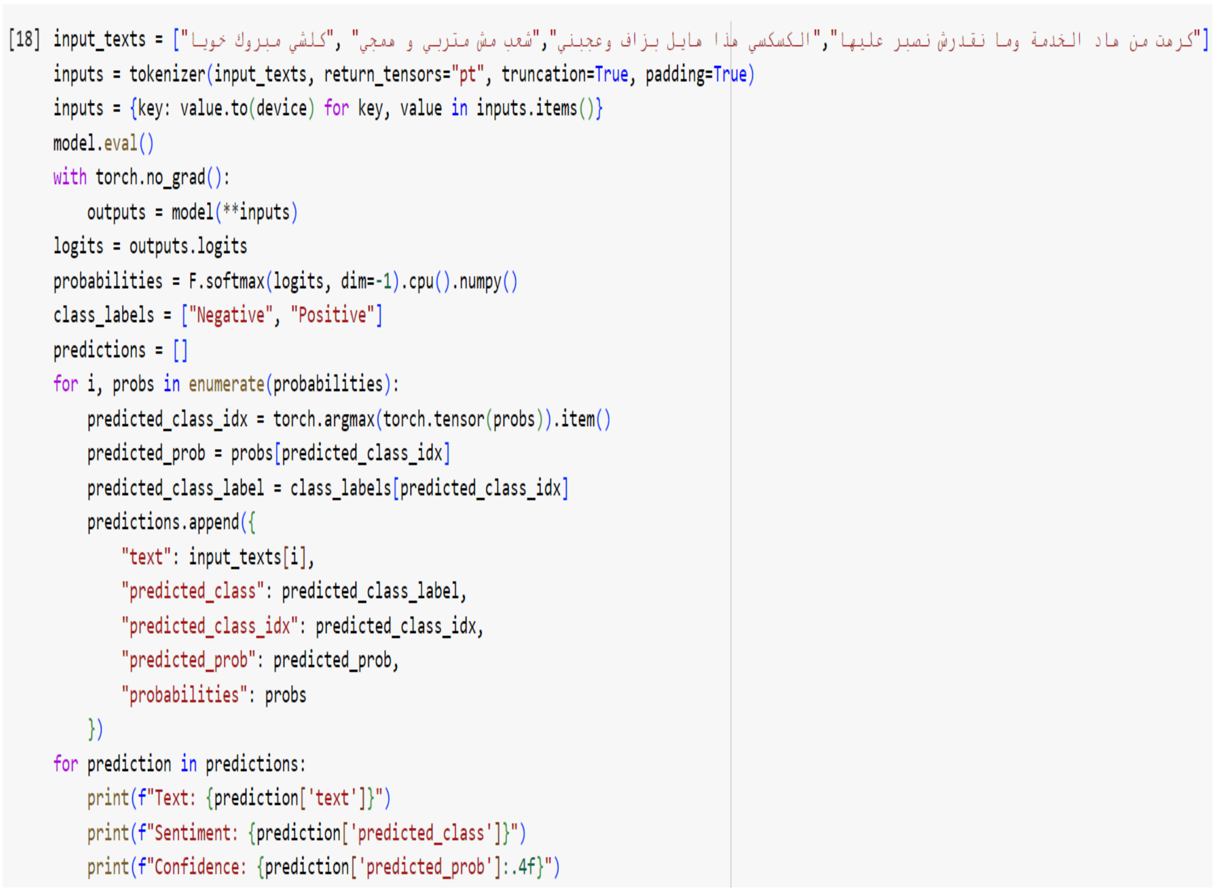
In Figure 3.10, we illustrate the detailed configuration of our LSTM model architecture.



```python
# Create LSTM model
lstm_model = Sequential()
lstm_model.add(Embedding(input_dim=vocab_size, output_dim=embedding_matrix.shape[1],
                         weights=[embedding_matrix], input_length=max_length, trainable=False))
# Add LSTM layers with dropout
lstm_model.add(LSTM(units=128, return_sequences=True))
lstm_model.add(Dropout(0.2))
lstm_model.add(LSTM(units=128))
lstm_model.add(Dropout(0.2))
lstm_model.add(Dense(1, activation='sigmoid'))  # Assuming 3 classes for sentiment analysis
# Custom learning rate
custom_lr = 0.001
# Compile the model with Adam optimizer and custom learning rate
lstm_model.compile(optimizer=Adam(learning_rate=custom_lr), loss='binary_crossentropy', metrics=['accuracy', Precision(), Recall(), f1_score])
```

Figure 3.10 – LSTM Model Architecture.

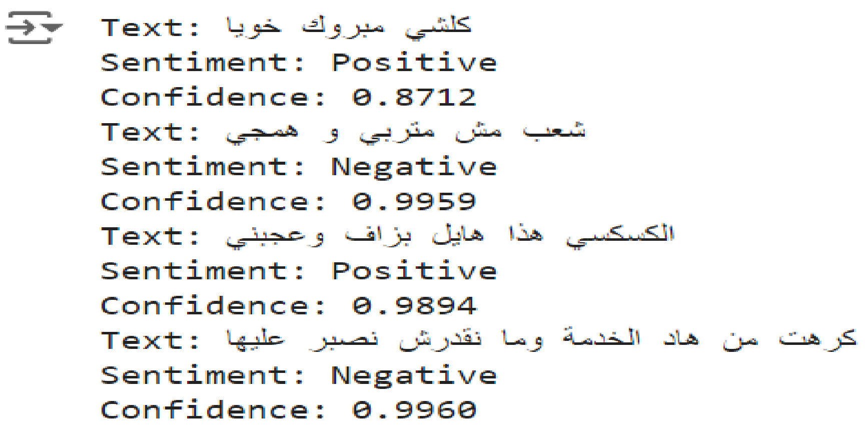
In Figure 3.11, we demonstrate the Arabic Bert base model inference.



```
[18] input_texts = ["كلشي مبروك خويا", "شعب مش متربي و همجي","الكسكسي هذا هايل بزاف وعجبني","كرهت من هاد الخدمة وما نقدرش نصبر عليها"]
     inputs = tokenizer(input_texts, return_tensors="pt", truncation=True, padding=True)
     inputs = {key: value.to(device) for key, value in inputs.items()}
     model.eval()
     with torch.no_grad():
         outputs = model(**inputs)
     logits = outputs.logits
     probabilities = F.softmax(logits, dim=-1).cpu().numpy()
     class_labels = ["Negative", "Positive"]
     predictions = []
     for i, probs in enumerate(probabilities):
         predicted_class_idx = torch.argmax(torch.tensor(probs)).item()
         predicted_prob = probs[predicted_class_idx]
         predicted_class_label = class_labels[predicted_class_idx]
         predictions.append({
             "text": input_texts[i],
             "predicted_class": predicted_class_label,
             "predicted_class_idx": predicted_class_idx,
             "predicted_prob": predicted_prob,
             "probabilities": probs
         })
     for prediction in predictions:
         print(f"Text: {prediction['text']}")
         print(f"Sentiment: {prediction['predicted_class']}")
         print(f"Confidence: {prediction['predicted_prob']:.4f}")
```

Figure 3.*11* – Arabic Bert base model Inference.

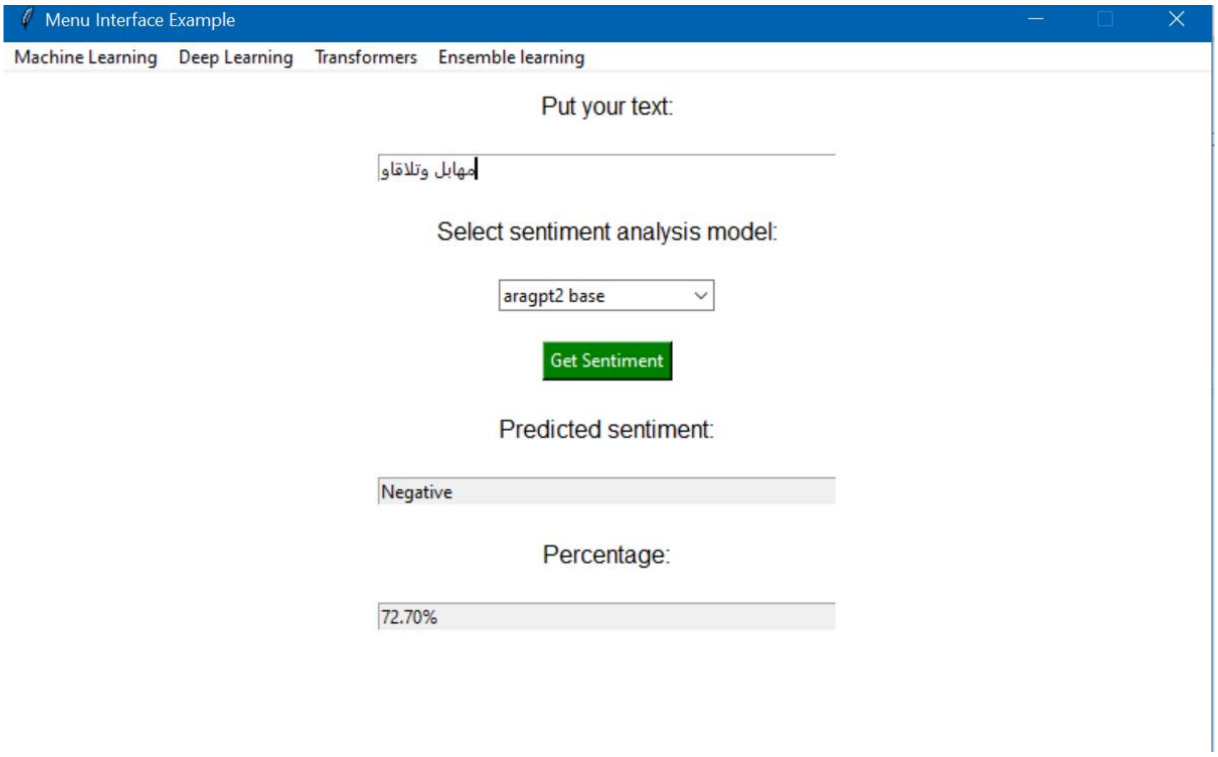
In Figure 3.12, we depict the obtained results by our Arabic Bert base model.



```
Text: كلشي مبروك خويا
Sentiment: Positive
Confidence: 0.8712
Text: شعب مش متربي و همجي
Sentiment: Negative
Confidence: 0.9959
Text: الكسكسي هذا هايل بزاف وعجبني
Sentiment: Positive
Confidence: 0.9894
Text: كرهت من هاد الخدمة وما نقدرش نصبر عليها
Sentiment: Negative
Confidence: 0.9960
```

Figure 3.*12* – Results obtained by Arabic Bert base model.

In Figure 3.13, we present the obtained results through our custom interface.



Put your text:

مهابل وتلاقاو

Select sentiment analysis model:

aragpt2 base

Predicted sentiment:

Negative

Percentage:

72.70%

Figure 3.13 – Results obtained through our developed interface.

3.6 Conclusion

In this chapter, we have elaborated the necessary tools required to realize our sentiment analysis experimental, through a few code sources presentation of our models. Furthermore, we have described the different models employed in our proposed approach on a dataset comprising 89,118 Algerian dialect comments, evenly labeled with 44,559 positive and 44,559 negative sentiments. When comparing the results obtained from our models, we notice that the SVM classifier attains the highest accuracy among machine learning classifiers, reaching 82%. Conversely, the highest accuracy obtained with the deep learning classifier was 85% through the LSTM model. Among the BERT models, the Arabic base architecture achieves an accuracy of 87.9% after an hour and 18 minutes of training time, emerging as the top-performing model in our experiment. Additionally, the ensemble learning methods yielded even higher accuracies, with both majority voting and stacking achieving accuracies of 90% and 91.1%, respectively, further enhancing the overall performance. However, it's not without its challenges, notably its struggle in classifying comments inscribed in Latin letters and its reliance on substantial labeled texts.

General Conclusion

To conclude, this master thesis has provided a comprehensive analysis of various classifiers for binary sentiment analysis, focusing on the Algerian dialect in Arabic characters. By comparing transformer-based models (BERT and GPT), traditional machine learning models (SVM, KNN, and Naive Bayes), deep learning models (LSTM and CNN), and exploring ensemble learning techniques such as majority voting and stacking, we aimed to identify the models or combinations thereof that offer optimal performance for sentiment analysis tasks involving complex and diverse datasets.

The findings revealed that ensemble learning methods that employed aragpt2, arabicbert base, and distilbert, along with transformer-based models, notably BERT, attained superior accuracy and robustness, surpassing traditional machine learning and deep learning models by a significant margin. This can be attributed to their sophisticated architectures and ability to capture complex linguistic patterns. Traditional models, while faster and less resource-intensive, struggled with the unique characteristics of the Algerian dialect, highlighting the importance of advanced preprocessing techniques and extensive dataset curation.

For future work, it is recommended to pre-train BERT models on larger datasets that include both Latin and Arabic characters to better capture the full range of linguistic features present in the Algerian dialect. Additionally, to optimize the training time of these models, replacing the ANN layer in BERT models with an ELM layer or a CNN layer could be explored. The ELM layer may offer faster training times, while the CNN layer could improve the model's ability to capture intricate patterns in the language, thereby enhancing overall performance.

Bibliography

[1] Liu, B. (2012). Sentiment analysis and opinion mining. Synthesis lectures on human language technologies.Morgan & Calypool Publishers, pp 1–167.

[2] Padmaja, S., & Fatima, S. S. (2013). Opinion mining and sentiment analysis-an assessment of peoples' belief: A survey. *International Journal of Ad hoc, Sensor & Ubiquitous Computing*, *4*(1), 21.

[3] Sudhir, P., & Suresh, V. D. (2021). Comparative study of various approaches, applications and classifiers for sentiment analysis. *Global Transitions Proceedings*, *2*(2), 205-211.

[4] Budiarsa, I. M. (2015). Language, dialect and register sociolinguistic perspective. *RETORIKA: Jurnal Ilmu Bahasa*, *1*(2), 379-387.

[5] Crystal, D., & Ivić, P. (2024, March 14). Dialect | Linguistics, Regional Variations and Dialectology. Encyclopedia Britannica. Https://www.britanica.com/topic/dialect.

[6] Saadane, H., & Habash, N. (2015). A conventional orthography for Algerian Arabic. In *the Second Workshop on Arabic Natural Language Processing* (pp. 69-79).

[7] Alwakid, G., Osman, T., & Hughes-Roberts, T. (2017). Challenges in sentiment analysis for Arabic social networks. *Procedia Computer Science*, *117*, 89-100.

[8] Moudjari, L., Akli-Astouati, K., & Benamara, F. (2020, May). An Algerian corpus and an annotation platform for opinion and emotion analysis. In *Proceedings of the Twelfth Language Resources and Evaluation Conference* (pp. 1202-1210).

[9] Thakkar, H., & Patel, D. (2015). Approaches for sentiment analysis on twitter: A state-of-art study. *Arxiv preprint arxiv:1512.01043*.

[10] Madhoushi, Z., Hamdan, A. R., & Zainudin, S. (2015, July). Sentiment analysis techniques in recent works. In *2015 science and information conference (SAI)* (pp. 288-291). IEEE.

[11] Udousoro, I.C., (2020). Machine learning: a review. *Semiconductor Science and Information Devices*, *2*(2), pp.5-14.

[12] Birjali, M., Kasri, M., & Beni-Hssane, A. (2021). A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems*, *226*, 107134.

[13] Jain, A. P., & Dandannavar, P. (2016, July). Application of machine learning techniques to sentiment analysis. In *2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (icatcct)* (pp. 628-632). IEEE.

[14] Gholami, R., & Fakhari, N. (2017). Support vector machine: principles, parameters, and applications. In *Handbook of neural computation* (pp. 515-535). Academic Press.

[15] Borele, P., & Borikar, D. A. (2016). An approach to sentiment analysis using artificial neural network with comparative analysis of different techniques. *IOSR Journal of Computer Engineering (IOSR-JCE)*, *18*(2), 64-69.

[16] Kukreja, H., Bharath, N., Siddesh, C. S., & Kuldeep, S. (2016). An introduction to artificial neural network. *Int J Adv Res Innov Ideas Educ*, *1*(5), 27-30.

[17] Parveen, H., & Pandey, S. (2016, July). Sentiment analysis on Twitter Data-set using Naive Bayes algorithm. In *2016 2nd international conference on applied and theoretical computing and communication technology (icatcct)* (pp. 416-419). IEEE.

[18] Ruz, G. A., Henríquez, P. A., & Mascareño, A. (2020). Sentiment analysis of Twitter data during critical events through Bayesian networks classifiers. *Future Generation Computer Systems*, *106*, 92-104.

[19] Jindal, K., & Aron, R. (2021). WITHDRAWN: A systematic study of sentiment analysis for social media data. *Materialstoday: PROCEEDINGS.*

[20] Bayhaqy, A., Sfenrianto, S., Nainggolan, K., & Kaburuan, E. R. (2018, October). Sentiment analysis about E-commerce from tweets using decision tree, K-nearest neighbor, and naïve bayes. In *2018 international conference on orange technologies (ICOT)* (pp. 1-6). IEEE.

[21] Hota, S., & Pathak, S. (2018). KNN classifier based approach for multi-class sentiment analysis of twitter data. *Int. J. Eng. Technol*, *7*(3), 1372-1375.

[22] Madhoushi, Z., Hamdan, A. R., & Zainudin, S. (2015, July). Sentiment analysis techniques in recent works. In *2015 science and information conference (SAI)* (pp. 288-291). IEEE.

[23] Ozgur, A. (2004). Supervised and unsupervised machine learning techniques for text document categorization. *Unpublished Master's Thesis, İstanbul: Boğaziçi University*.

[24] Reddy, Y. C. A. P., Viswanath, P., & Reddy, B. E. (2018). Semi-supervised learning: A brief review. *Int. J. Eng. Technol*, *7*(1.8), 81.

[25] Uc-Cetina, V., Navarro-Guerrero, N., Martin-Gonzalez, A., Weber, C., & Wermter, S. (2023). Survey on reinforcement learning for language processing. *Artificial Intelligence Review*, *56*(2), 1543-1575.

[26] Wang, X., Jiang, W., & Luo, Z. (2016, December). Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In *Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers* (pp. 2428-2437).

[27] Basiri, M. E., Nemati, S., Abdar, M., Cambria, E., & Acharya, U. R. (2021). ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Generation Computer Systems, 115*, 279-294.

[28] Shen, Q., Wang, Z., & Sun, Y. (2017). Sentiment analysis of movie reviews based on cnn-blstm. In *Intelligence Science I: Second IFIP TC 12 International Conference, ICIS*

2017, Shanghai, China, October 25-28, 2017, Proceedings 2 (pp. 164-171). Springer International Publishing.

[29] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

[30] Safaya, A., Abdullatif, M., & Yuret, D. (2020). Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media. *Arxiv preprint arxiv:2007.13184*.

[31] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). Distilbert, a distilled version of BERT: Smaller, faster, cheaper and lighter. Arxiv 2019. *Arxiv preprint arxiv:1910.01108*.

[32] Antoun, W., Baly, F., & Hajj, H. (2020). Aragpt2: Pre-trained transformer for Arabic language generation. *Arxiv preprint arxiv:2012.15520*.

[33] Wankhade, M., Rao, A. C. S., & Kulkarni, C. (2022). A survey on sentiment analysis methods, applications, and challenges. *Artificial Intelligence Review*, *55*(7), 5731-5780.

[34] Zhou, Z. H. (2012). *Ensemble methods: foundations and algorithms*. CRC press.

[35] Mohammed, A., & Kora, R. (2023). A comprehensive review on ensemble deep learning: Opportunities and challenges. *Journal of King Saud University-Computer and Information Sciences*, *35*(2), 757-774.

[36] Abdaoui, A., Berrimi, M., Oussalah, M., & Moussaoui, A. (2021). Dziribert: a pre-trained language model for the algerian dialect. *Arxiv preprint arxiv:2109.12346*.

[37] Benmounah, Z., Boulesnane, A., Fadheli, A., & Khial, M. (2023). Sentiment Analysis on Algerian Dialect with Transformers. *Applied Sciences*, *13*(20), 11157.

[38] Mazari, A. C., & Djeffal, A. (2022). Sentiment analysis of Algerian dialect using machine learning and deep learning with Word2vec. *Informatica*, *46*(6).

[39] Abdelli, A., Guerrouf, F., Tibermacine, O., & Abdelli, B. (2019, December). Sentiment analysis of Arabic Algerian dialect using a supervised method. In *2019 international conference on intelligent systems and advanced computing sciences (ISACS)* (pp. 1-6). IEEE.

[40] Chader, A., Lanasri, D., Hamdad, L., Belkheir, M. C. E., & Hennoune, W. (2019, December). Sentiment Analysis for Arabizi: Application to Algerian Dialect. In *KDIR* (pp. 475-482).

[41] Guellil, I., Adeel, A., Azouaou, F., & Hussain, A. (2018). Sentialg: Automated corpus annotation for algerian sentiment analysis. In *Advances in Brain Inspired Cognitive Systems: 9th International Conference, BICS 2018, Xi'an, China, July 7-8, 2018, Proceedings 9* (pp. 557-567). Springer International Publishing.

[42] Rahab, H., Zitouni, A., & Djoudi, M. (2021). SANA: Sentiment analysis on newspapers comments in Algeria. *Journal of King Saud University-Computer and Information Sciences*, *33*(7), 899-907.

[43] Soumeur, A., Mokdadi, M., Guessoum, A., & Daoud, A. (2018). Sentiment analysis of users on social networks: Overcoming the challenge of the loose usages of the Algerian dialect. *Procedia computer science*, *142*, 26-37.

[44] Guellil, I., Adeel, A., Azouaou, F., Benali, F., Hachani, A. E., Dashtipour, K., ... & Hussain, A. (2021). A semi-supervised approach for sentiment analysis of arab (ic+ izi) messages: Application to the algerian dialect. *SN Computer Science*, *2*, 1-18.

[45] Hamadouche, K., Bousmaha, K. Z., Bekkoucha, M. A., & Hadrich-Belguith, L. (2023). Algbert: Automatic Construction of Annotated Corpus for Sentiment Analysis in Algerian Dialect. *ACM Transactions on Asian and Low-Resource Language Information Processing*, *22*(12), 1-17.

[46] Kewsuwun, N., & Kajornkasirat, S. (2022). A sentiment analysis model of agritech startup on Facebook comments using naive Bayes classifier. *International Journal of Electrical & Computer Engineering (2088-8708)*, *12*(3).

[47] Touileb, S., & Barnes, J. (2021). The interplay between language similarity and script on a novel multi-layer Algerian dialect corpus. *Arxiv preprint arxiv:2105.07400*.

[48] Hakim, A. A., Erwin, A., Eng, K. I., Galinium, M., & Muliady, W. (2014, October). Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach. In *2014 6th international conference on information technology and electrical engineering (ICITEE)* (pp. 1-4). IEEE.

[49] Rana, R., & Singhal, R. (2015). Chi-square test and its application in hypothesis testing. *Journal of Primary Care Specialties*, *1*(1), 69-71.

[50] Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: an overview. *Arxiv preprint arxiv:2008.05756*.

[51] Teoh, T. T., & Rong, Z. (2022). *Artificial intelligence with python* (pp. 3-328). New York: Springer.

[52] Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with numpy. *Nature*, *585*(7825), 357-362.

[53] mckinney, W. (2010, June). Data structures for statistical computing in Python. In *scipy* (Vol. 445, No. 1, pp. 51-56).

[54] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, *9*(03), 90-95.

[55] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *The Journal of machine Learning research*, *12*, 2825-2830.

[56] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Zheng, X. (2016). {tensorflow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)* (pp. 265-283).

[57] Ketkar, N., & Ketkar, N. (2017). Introduction to keras. *Deep learning with python: a hands-on introduction*, 97-111.

[58] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems, 32*.

[59] *Python Software Foundation. (n.d.). Re — Regular expression operations. Available at: https://docs.python.org/3/library/re.html.*

[60] *Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media, Inc. Available at: https://www.nltk.org/book/.*

[61] *Zerrouki, T. (n.d.). Pyarabic. Github. Available at: https://github.com/linuxscout/pyarabic.*

[62] *Python Software Foundation. (n.d.). Os — Miscellaneous operating system interfaces. Available at: https://docs.python.org/3/library/os.html.*

[63] *Python Software Foundation. (n.d.). Json — JSON encoder and decoder. Available at: https://docs.python.org/3/library/json.html.*

[64] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020, October). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations* (pp. 38-45).

[65] Lundh, F. (1999). An introduction to tkinter. *URL: www. Pythonware. Com/library/tkinter/introduction/index. Htm*.

[66] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., ... & Willing, C. (2016). Jupyter Notebooks-a publishing format for reproducible computational workflows. *Elpub*, *2016*, 87-90.

[67] Mataoui, M. H., Zelmati, O., & Boumechache, M. (2016). A proposed lexicon-based sentiment analysis approach for the vernacular Algerian Arabic. *Research in Computing Science*, *110*(1), 55-70.