Thesis submitted to the

UNIVERSITY OF MOHAMED BOUDIAF – MSILA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in **Computer science**

By

BEDDAR Mohammed Iqbal

Title of the thesis

Algerian dialect sentiment analysis in social networks

Under the supervision of

Dr Marouane KIHAL

Composition of the jury

Rabah Mokhtari	University of Msila	President
Marouane KIHAL	University of Msila	Reporter
Nour elhouda chalabi	University of Msila	Examiner

06, 2026

Abstract

This dissertation aims to explore the field of sentiment analysis as a key application of Natural Language Processing (NLP), with a special emphasis on the Arabic language and the Algerian dialect due to their linguistic diversity and the challenges they pose for automatic processing and comprehension. It covers the various stages of sentiment analysis,

followed by preprocessing, feature extraction, and the development of classification models. The work also underscores the importance of NLP in understanding Arabic texts and examines the historical and cultural factors that have shaped the emergence and development of the Algerian dialect. From a practical standpoint, the study presents a sentiment analysis model and compares its results and performance with several other models to assess its effectiveness and accuracy in classifying sentiments within texts written in the Algerian dialect.

Key words: sentiment analysis , Algerian dialect , model , Natural Language Processing, machine learning , deep learning

---

الملخص

تهدف هذه المذكرة إلى دراسة مجال تحليل المشاعر كأحد تطبيقات معالجة اللغات الطبيعية، مع التركيز بشكل خاص على اللغة العربية واللهجة الجزائرية لما تتميز به من تنوع لغوي وصعوبات في المعالجة والفهم الآلي. تتناول المذكرة مراحل تحليل المشاعر . مرورًا باستخراج الخصائص، وانتهاءً ببناء نماذج التصنيف. كما تستعرض أهمية معالجة اللغات الطبيعية في فهم النصوص العربية، مع تسليط الضوء على العوامل التاريخية والثقافية التي أثرت في نشأة اللهجة الجزائرية وتطورها. من الناحية التطبيقية، تقدم الدراسة نموذجًا لتحليل المشاعر، مع مقارنة نتائجه وأداءه بعدة نماذج أخرى لتقييم فعاليته ودقته في تصنيف المشاعر داخل النصوص باللهجة الجزائرية.

الكلمات المفتاحية : تحليل المشاعر ، اللهجة الجزائرية ، نموذج ، معالجة اللغات الطبيعية ،تعلم الآلة ، التعلم العميق

Dedications

"I dedicate this work to my family, whose unwavering support has sustained me throughout my academic journey. I also express my deepest gratitude to the teaching staff who have guided and educated me, enabling me to reach this stage. Special thanks are extended to my supervisor, whose invaluable assistance and mentorship have been instrumental in my research and work. Additionally, I am grateful to my university friends, who stood by me during times of difficulty and provided constant encouragement"

beddar Mohamed Ikbal

Acknowledgments

I am deeply indebted to my thesis supervisor Marouane kihal whose unlimited steadfast support and inspirations have made this project a great success. In a very special way. I thank him for every support he has rendered unto me to see that i succeed in this challenging study.

Special thanks go to my friends and families who have contained the hectic moments and stress i have been through during the course of there search project.

Contents

List of figures

List of tables

List of Abbreviations

AI : Artificial Intelligence

DL : Deep Learning

ML : Machine Learning

NLP : Natural Language Processing

SA : Sentiment analysis

KNN : K-nearest neighbors

RF : Random Forest

DT : Decision Trees

CNN : Convolutional Neural Networks

RNN : Recurrent Neural Networks

LSTM : Long Short-Term Memory

GRU : Gated Recurrent Units

BERT : Bidirectional Encoder Representations from Transformers

BILSTM : Bidirectional Long Short-Term Memory

SMOTE : Synthetic Minority Over-sampling Technique

Introduction

Researchers in the field of computer science trace the origins of Artificial Intelligence (AI) to early 1955, when John McCarthy approached the Rockefeller Foundation to request funding for a summer workshop at Dartmouth College, intended for approximately ten participants. In the summer of 1956, McCarthy, together with Claude Shannon one of the founders of information theory at Bell Labs met with Robert Morison, Director of Biological and Medical Research, to discuss the concept and potential funding. Although Morison expressed skepticism regarding the feasibility of financing such an ambitious project, the initiative proceeded. [1]

On September 2, 1955, the formal project proposal was submitted by McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon. [2,3] This proposal, known as the Dartmouth Summer Research Project on AI, is credited with coining the term "Artificial Intelligence." The project aimed to conduct a study in the summer of 1956 at Dartmouth College in Hanover, New Hampshire, based on the premise that every aspect of learning or other features of intelligence could, in principle, be described with sufficient precision to enable machine simulation. The researchers sought to investigate how machines could utilize language, form abstract concepts, solve problems currently solvable only by humans, and improve their own functioning. They hypothesized that significant advancements could be realized if a carefully selected group of scientists collaborated intensively over the summer.

Today, AI is recognized as one of the most transformative technological advancements of the 21st century. Both developed and developing nations have demonstrated considerable interest in expanding AI research and allocating substantial financial resources to it, given its critical role in accelerating development, advancing scientific inquiry, and addressing complex global challenges.

In essence, Artificial Intelligence refers to the capacity of machines to simulate human cognitive functions in executing complex tasks. AI has become an integral component of modern society, evident in applications ranging from smartphones and healthcare to industrial processes. It offers numerous benefits, including increased productivity, enhanced accuracy in medical diagnostics, and the facilitation of everyday

activities through virtual assistants such as Siri and ChatGPT. Furthermore, AI systems can process vast quantities of data at speeds unattainable by humans.

Conversely, the proliferation of AI technology raises significant concerns. Automation threatens to displace a substantial portion of the human workforce, potentially exacerbating unemployment rates. Additionally, the development of autonomous weaponry and the potential misuse of personal data present profound ethical challenges. Thus, AI constitutes a dual-edged phenomenon, necessitating stringent regulatory frameworks to ensure its deployment benefits humanity and mitigates potential harms.

Research Objectives:

The field of Artificial Intelligence has expanded to include the humanities, linguistics, and dialectology, which is a branch of linguistics concerned with studying language variation particularly geographical differences in pronunciation, grammar, and vocabulary. This research falls within the framework of developing a program that analyzes human sentiment in Algerian dialects using Natural Language Processing (NLP) techniques within AI.

The aim is to identify the most accurate model capable of understanding human emotions based on local dialects. The program analyzes sentiments expressed in comments, opinions, tweets, written texts, as well as facial expressions and emotional reactions. It also processes content shared on social media platforms (such as Facebook, TikTok, and X), evaluating it in terms of positivity and negativity.

Through this model, researchers can propose solutions in cases of negative sentiment or maintain and improve conditions in cases of positive sentiment. For example, public opinion can be analyzed through people's comments on Facebook regarding the price of sacrificial animals during Eid al-Adha. If the sentiment is predominantly negative, authorities can take appropriate measures to ease the burden on citizens, such as increasing livestock imports.

In general, the objective of this research is to transform human emotions into actionable data that can be used to design and support developmental, health, social, or entertainment policies.

The rapid expansion of the internet and social media platforms has profoundly transformed the ways in which individuals communicate and express their opinions. Millions of users share their emotions, thoughts, and reactions daily through comments, reviews, tweets, and online discussions, generating a substantial volume of textual data. This continuous flow of information has become a valuable resource for researchers, corporations, and institutions aiming to gain deeper insights into public opinion and human behavior.

Within this context, sentiment analysis has emerged as a pivotal application of Natural Language Processing (NLP). It involves the automatic identification and classification of emotions and opinions expressed in text into categories such as positive, negative, or neutral. Unlike traditional data analysis methods that predominantly focus on numerical data, sentiment analysis seeks to interpret the subjective and emotional dimensions inherent in human language.

The significance of sentiment analysis spans multiple domains. In business and marketing, organizations utilize it to assess customer satisfaction and refine products and services based on user feedback. In healthcare, sentiment analysis aids in detecting emotional distress and understanding psychological states through textual interactions. Similarly, governments and public institutions employ these techniques to gauge public sentiment regarding social, economic, or political matters. Social media platforms also leverage sentiment analysis to identify harmful or sensitive content, thereby enhancing user experience.

Despite its growing importance, sentiment analysis remains a complex challenge due to the inherent complexity, ambiguity, and contextual variability of human language. These challenges are further amplified when analyzing the Arabic language and its dialects, particularly the Algerian dialect, which is marked by significant linguistic diversity and the influence of multiple languages including Amazigh, French, Turkish, and Spanish. Additionally, social media users frequently employ informal writing styles, abbreviations, emojis, and code-switching between languages, complicating automated text comprehension.

Therefore, the development of accurate sentiment analysis systems tailored to the Algerian dialect has become a critical area of research within AI and Natural Language Processing. Recent advances in machine learning, deep learning, and transformer-based architectures have considerably improved computational systems' capabilities to process dialectal language and extract meaningful emotional insights from textual data.

Research Structure:

Based on the research idea, we have structured this study as follows:

- **General Introduction :**

In This section, we will provide a brief overview of Artificial Intelligence how it began, how it has evolved, its diversity, and its integration into various technical, scientific, human, and medical fields, as well as its presence in all aspects of human life.

- **Chapter One :**

This chapter will address the definition of sentiment analysis and Algerian dialects, along with the factors influencing them, such as geographical and historical elements, and the role of customs and traditions. It will also includes an introduction to Artificial Intelligence in the context of Natural Language Processing (NLP), which will serves as a foundation for the research objective namely, sentiment analysis through specific models.

- **Chapter two :**

In this chapter, we will explore various models used for sentiment analysis across Machine Learning, Deep Learning, and Transformer architectures. We will cover their operational methods and present experimental examples showcasing the results obtained from their application.

- **Chapter Three :**

In this chapter, we will discuss the stages of sentiment analysis and the key concepts underlying these stages. We will define each stage, explained its characteristics and features, and presented the model architecture that we have developed.

- **Chapter Four:**

This chapter will present the results obtained from applying the previously mentioned models, as well as the outcomes of our proposed model. It will also includes a detailed explanation of the model’s structure.

Chapter 1 Sentiment Analysis for the Algerian Dialect Using Natural Language Processing

1.1. Introduction:

Artificial intelligence has had a significant impact in our time, as it is one of the most important current investments that major companies are working on. It has helped humans in several areas, such as financial, technical, mechanical, and programming problems. The field of AI has developed in recent years due to human interest in this science and has branched into several subfields, ultimately including the field that was created to help humans, known as Natural Language Processing, which helped strengthen the relationship between humans and machines. Humans and companies have particularly benefited from it in understanding the opinions and feelings of their customers and improving the relationship between companies and consumers across all languages and writing styles.

1.2. The Concept of Sentiment Analysis:

Sentiment analysis, also known as opinion mining, is the process of analyzing large volumes of text to determine a person's feelings or opinions whether they are positive, negative, or neutral. Companies benefit from this technology by improving their relationships with customers through analyzing their opinions and emotions toward products and services, and by making future strategic decisions aligned with their goals. [4]

Information is collected from various sources such as tweets, comments, chat messages, and emails across different social media platforms (Facebook, X, TikTok, etc.). Sentiment analysis aims to understand users' emotions, recognize their psychological state, and ensure their well-being. For this reason, some social media platforms display supportive or solidarity messages when users search for negative or sensitive terms.

Additionally, sentiment analysis can determine whether a written comment is positive, neutral, or negativesometimes helping platforms decide whether content should be moderated or removed. Through this system, organizations can better

understand users' emotions and extract customers' opinions in order to improve customer service and enhance brand reputation.

In the past, the study of human opinions and emotions was limited due to the scarcity of texts containing personal views before the emergence of the internet. Individuals used to consult their family members and friends, while institutions relied on opinion polls, surveys, and focus groups. However, with the advent of the internet especially with the ma

ssive growth of visual and audio content in recent years the world has changed dramatically. It has transformed the way people express their opinions, as they can now publish product reviews on e-commerce websites and share their views about almost anything on social media platforms .

Emotion appears in text in two forms: an explicit form, where a subjective sentence clearly expresses an opinion (e.g., "It is a beautiful day"), which is often the focus since it is easier to analyze; and an implicit form, where the text only suggests an opinion (e.g., "The earphones broke within two days"). Therefore, after processing the texts and analyzing all sentences both explicit and implicit based on a database, we can extract and determine their position on a spectrum and identify whether they lean toward a negative or positive opinion.

1.3. The Algerian Dialect:

There are currently more than 7100 languages spoken around the world. Most languages are characterized by the presence of dialects that vary according to the nature of the people, their geographical location, and their history. For example, in the Arabic language, there are Gulf dialects such as Saudi, Kuwaiti, Qatari, and Yemeni, as well as the Iraqi dialect, which itself branches into several sub-dialects (northern and southern). There are also Egyptian dialects and many other varieties of Arabic. In this research, we will focus on the Algerian dialect. [5]

The Algerian dialect, and Maghrebi dialects in general, differ significantly from other Arabic dialects due to differences in the origins of the populations, their contact with various civilizations, and the influence of their surrounding environment. For this

reason, it is often difficult for Arabs in the Middle East to understand Maghrebi dialects. In the case of Algeria, this can be attributed to three main factors:

1.3.1. Historical Factors:

The history of Algeria is very ancient. In the past, it was known as the kingdom of Numidia, whose main language was the Amazigh (Berber) language. Later, several civilizations ruled the region, including the Roman and Byzantine civilizations.

With the Islamic conquest and the spread of the Arabic language for reading the Holy Qur'an, the majority of the population began speaking Arabic while still preserving their local languages[6].

Algerians were also influenced by contact with many peoples, whether through trade exchanges or through military incursions affecting northern Algeria. Furthermore, the presence of the Ottoman Empire and later the French Algeria contributed to enriching Algerian dialects.

As a result, the Algerian dialect became a mixture of Arabic, Amazigh, Turkish, Italian, Spanish, and French words, where French vocabulary in particular became widely used and represents a significant portion of everyday Algerian speech.

Below are some examples of words commonly known in the Algerian dialect :

mother language	The word
Amazigh languages	شلاغم (شوارب)
Arabic languages	أهلا
french language	بيرو (مكتب)
spanish language	سباط (حذاء)
turkish language	بلاك (ربما)

Table 1. the words and their source language

1.3.2. Geographical Factors:

The location and large size of Algeria have contributed to the distinctiveness of its dialect, even within the country itself. As mentioned in the historical factors, Algeria overlooks the Mediterranean Sea, which is one of the most important water bodies. This has allowed coastal provinces to be more exposed to foreign languages compared to southern provinces.

As show in [Fig 1.], Algeria's geographical location and its land borders with multiple neighboring countries have fostered a rich diversity of Algerian dialects, leading to distinct differences between the eastern and western regional dialects. This influence is evident in the similarities between certain local dialects and those of neighboring countries. Additionally, dialectal features can often be used to identify the specific state or region a speaker originates from.

Fig 1. Map of the Algerian borders

1.3.3. Human Factors:

Although the human brain has high intelligence and the ability to think and solve problems compared to many other creatures, it tends to favor mental efficiency and reduce effort. Over time, when people repeat words frequently, language tends to become simplified into forms that are easier to pronounce.

For example, in the English language, instead of saying or writing *"let me"*, many people shorten it to *"lemme"*. The same phenomenon occurs in Arabic and the Algerian dialect, where some words are simplified, such as *"مالك"*, which is derived from *" ماذا بك"*. In some cases, the word may change completely, such as *"موس"* in the Amazigh language, which means *"a lot"* instead of *"كثير"*, because it is easier for the speech muscles to pronounce. The diagram in the [fig 2.] explains how Arabic dialects function to influence the cuisine of an Arabic-speaking country.

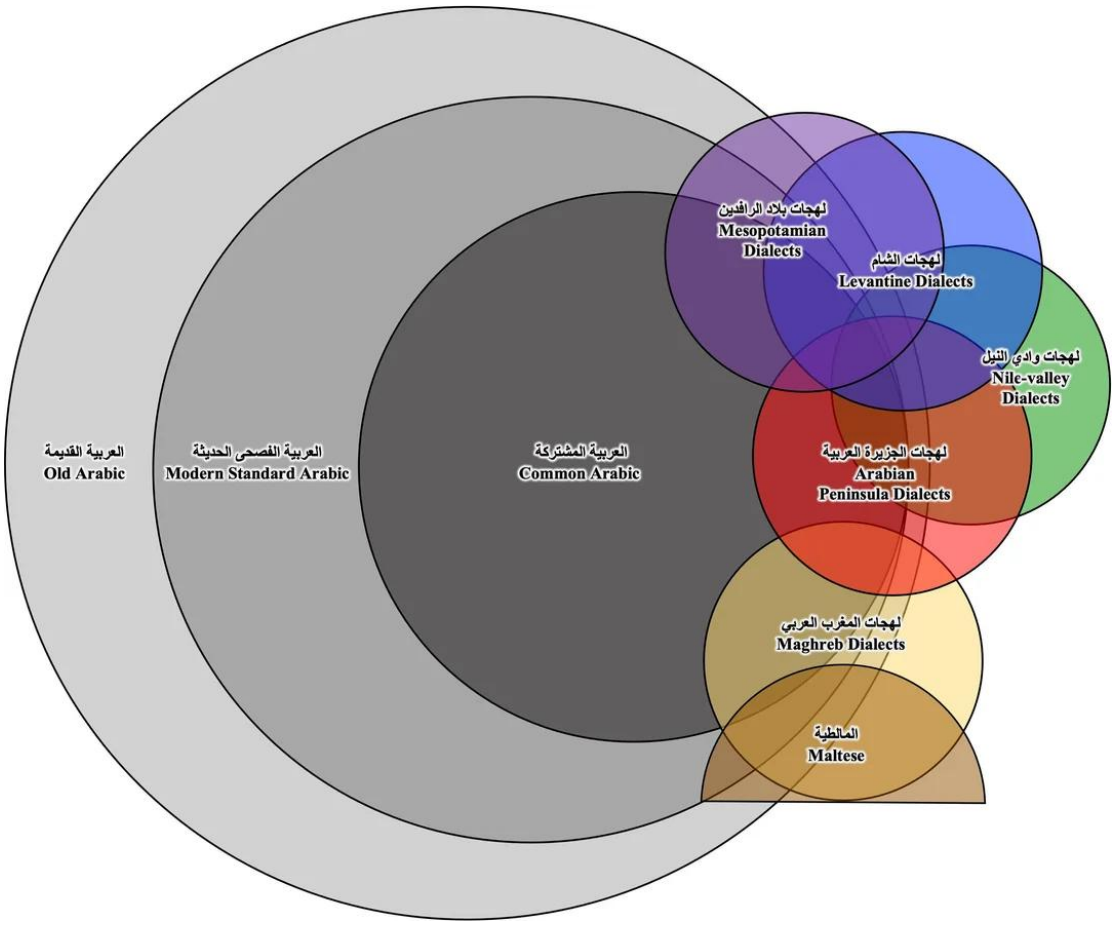

Fig 2. A Venn Diagram of Arabic Dialects

1.4. The Concept of Natural Language Processing (NLP) :

Natural Language Processing (NLP) is a core subfield of Artificial Intelligence dedicated to enabling machines to comprehend, interpret, and generate human language in a meaningful manner. It acts as an interface between human communication and computational understanding, facilitating the processing of both textual and spoken data in languages such as English, Arabic, French, and many others. The primary goal

of NLP extends beyond simple word recognition to encompass the capture of context, meaning, intent, and the linguistic subtleties inherent in natural language.

In practice, NLP confronts the intrinsic complexity of human language, which is often ambiguous, unstructured, and heavily context-dependent. Features such as idiomatic expressions, sarcasm, dialectal variation, and cultural nuances pose significant challenges for automated language understanding. To overcome these obstacles, NLP systems integrate linguistic knowledge with statistical methodologies to model language in a manner that is both computationally efficient and semantically rich.

The evolution of NLP has progressed through multiple phases. Initial approaches were predominantly rule-based, relying on manually crafted grammatical rules and lexicons to process language. While effective within constrained environments, these systems lacked adaptability and struggled to manage the variability of natural language in real-world contexts. The advent of Machine Learning (ML) introduced a paradigm shift, enabling models to learn linguistic patterns directly from data rather than depending on handcrafted rules. This advancement facilitated the handling of larger datasets and more complex linguistic phenomena.

The emergence of Deep Learning (DL) further revolutionized NLP by allowing models to autonomously learn hierarchical representations of language. Modern architectures, including recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and transformers, have substantially enhanced performance across diverse linguistic tasks. These models excel at capturing sequential dependencies and contextual relationships between words, which are essential for accurate semantic interpretation.

Currently, NLP applications span numerous domains such as information retrieval, machine translation, text classification, conversational agents, and sentiment analysis. In the domain of sentiment analysis, NLP techniques extract subjective information from texts, categorizing opinions as positive, negative, or neutral. This capability is particularly valuable for analyzing social media content, customer feedback, and public opinion.

Despite significant advancements, NLP continues to face challenges, especially in processing low-resource languages, dialectal variants, and informal communication modes including code-switching and slang. These issues are notably pronounced in linguistically diverse languages like Arabic and its dialects, where contextual and regional differences greatly influence meaning.

In summary, NLP remains a dynamic and rapidly advancing field, playing a vital role in enhancing human–computer interaction and enabling machines to process and understand natural language with increasing precision and sophistication.[7]

1.5. Conclusion :

After understanding the concept of Natural Language Processing and how it works in interpreting human language and simplifying it for machine understanding, and after examining the origins of the Algerian dialect and the factors that influenced its formation, we can use psychological analysis to examine texts and extract emotions or opinions, whether positive or negative. This helps in making more accurate decisions on social media platforms by better understanding users' impressions and linguistic and psychological behavior

This leads us to the question of how these data can be integrated, and more precisely how Natural Language Processing can work with the Algerian dialect in practice. This is what we will discuss in the second chapter

Chapter 2 Artificial Intelligence Models for Sentiment analysis

2.1. Maching learning in sentiment analysis :

Machine learning in sentiment analysis involves using computational algorithms to automatically identify and classify opinions expressed in text. It enables systems to determine whether a piece of text such as a review, tweet, or comment conveys positive, negative, or neutral sentiment. Leveraging techniques from Natural Language Processing, machines process human language, extract meaningful features, and learn patterns from large datasets. Common models used include Naive Bayes, Support Vector Machines, and advanced deep learning approaches like neural networks.

This approach is vital in many real-world applications such as customer feedback analysis, social media monitoring, and market research. ML models improve with more training data, resulting in more accurate predictions and enhanced understanding of human emotions in text. Compared to traditional rule-based methods, ML offers greater flexibility and adaptability, making it a powerful and efficient tool for analyzing large volumes of unstructured textual data.

2.1.1KNN (K-nearest neighbors) :

k-Nearest Neighbors (KNN) is a non-parametric classification algorithm. Despite its simplicity, it is effective in many cases. When classifying a data instance *T*, its nearest neighbors are identified based on the value of *K*, forming a surrounding set around *T*. Typically, majority voting among these neighbors is used to determine the class of *T*, with or without considering distance weighting. As we see in the picture [Fig 3.]

The KNN algorithm depends heavily on the value of *K*, as the classification result is strongly influenced by it. There are several methods for selecting an appropriate *K*, but one of the simplest approaches is to run the algorithm multiple times using different values of *K*. It is generally recommended to use odd values of *K* (i.e., 2n + 1)

The dependence on *K* can be reduced by using the concept of contextual probability. This approach involves aggregating support from multiple groups of nearest neighbors belonging to different classes, resulting in a more reliable support value that better reflects the true class of *T*. However, despite its improved accuracy and reduced dependence on *K*, this method has a drawback: it is computationally slow, requiring a time complexity of $O(n^2)$ to classify a new instance.[8]

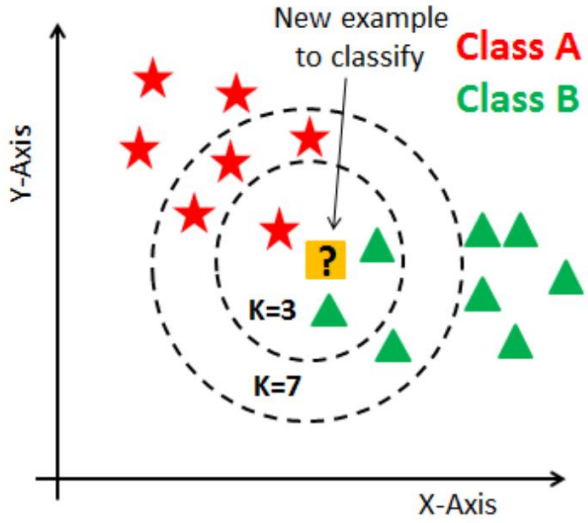

Fig 3. Format of KNN

KNN in sentiment analysis：

In sentiment analysis, the objective is to determine whether a given data instance is positive or negative, and in some cases, a neutral class is also included. This allows the use of the KNN algorithm due to its good performance, effectiveness, and simplicity. The unknown instance *T* is assigned to the most common class among its nearest neighbors. In the case of a tie between classes, the class with the smallest average distance to the unknown instance *T* is selected .

A study by H. Soudamini et al.[9] discussed a methodology that relies on the SVM algorithm to train a classifier for distinguishing Twitter data (now known as X). It also explained how this methodology was improved by using the KNN algorithm to train the classifier. The results showed a significant improvement by selecting an appropriate value of K, using a suitable distance metric, applying feature weighting, and removing noisy data to simplify and accelerate the process.

A comparison between the two approaches demonstrated that KNN outperformed SVM in terms of accuracy, precision, and recall. The study also indicated that KNN performance can be further improved by using Distance Weighted KNN.

2.1.2 Decision tree (DT) :

Decision trees are considered one of the most successful machine learning algorithms due to their characteristics, such as simplicity, ease of learning, no need for parameter tuning, and the ability to handle different types of data. A decision tree is built from a set of labeled training examples, where each example is represented by a set of attribute values along with a class label.

Decision trees typically use methods such as a greedy approach, a top-down strategy, and a recursive process. The algorithm starts with all the training data and an empty tree. It selects the attribute that best splits the training data to serve as the root node, then divides the data into separate subsets according to the values of that attribute. This process is then applied recursively to each subset until all examples within a subset belong to the same class. As shown in [fig 4.]

Among the advantages of decision trees are their ease of understanding and interpretation, as they do not require complex configuration to operate, and they work well with textual data after it has been transformed into features. However, they also have some disadvantages, including the tendency to become overly complex (overfitting), sensitivity to changes in the data, and in some cases, lower performance compared to other algorithms such as SVM [10]

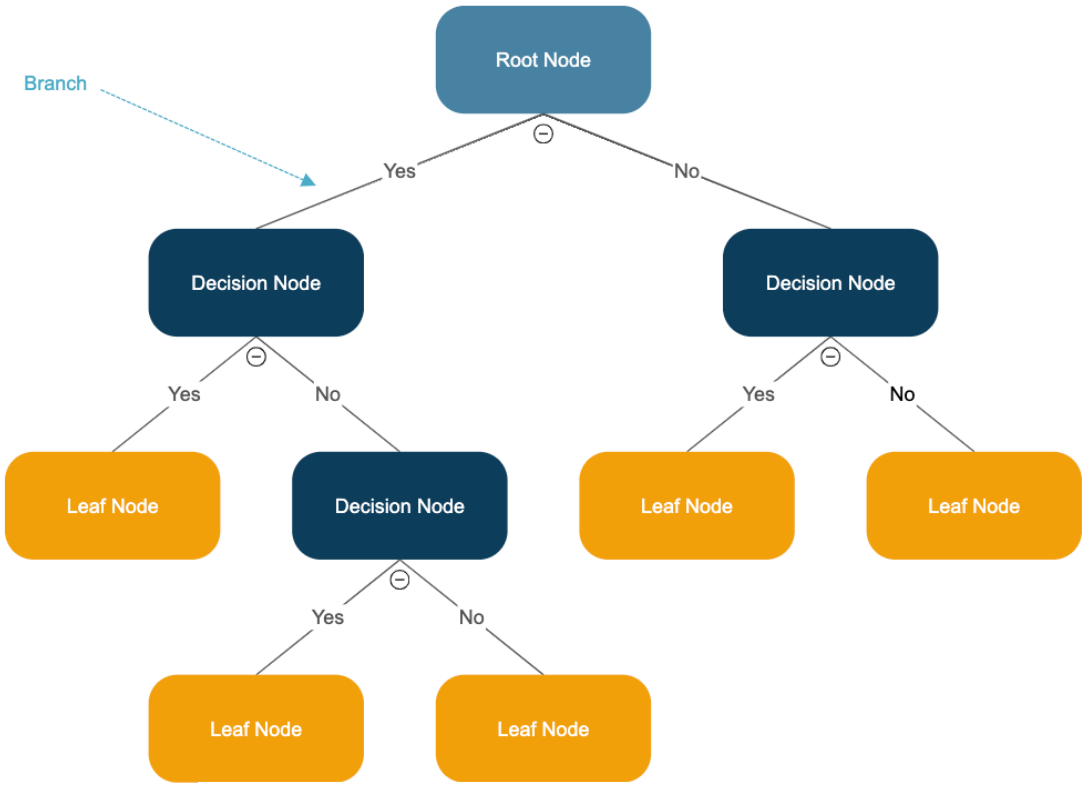

Fig 4. Format of decision tree

Dection tree in sentiment analysis：

A decision tree relies on splitting data or sentences in the case of text into features such as the presence of keywords (e.g “good” or “bad”), the number of positive and negative words, sentence length, and repeated words. The algorithm then selects the best feature to split the data and creates branches based on the values of that feature. This process continues recursively until the end of the tree is reached.

In a study by B. Achmad et al.[11], a comparison was conducted between three types of algorithms: KNN, Decision Tree, and Naïve Bayes. The study analyzed Twitter posts related to customer opinions about several e-commerce websites in Indonesia. The results showed that the Decision Tree achieved :

	accuracy	precision	Recall
Results	80%	79.96%	84%

2.1.3 Random forest (RF) :

Random Forest is one of the machine learning algorithms used in classification and prediction. It is based on the idea of combining multiple decision trees instead of using a single tree. This ensemble technique, known as bagging, works by combining several independent decision trees, where each tree is built using random samples of the data (which may be repeated) and a random subset of features.

When new data is introduced, each tree produces its own prediction, and these predictions are then aggregated. In classification tasks, the final class is chosen either through voting or by averaging the results in regression tasks. As illustrated in [fig 5]

Random Forest achieves high accuracy because it relies on the results of multiple trees, which also helps reduce the problem of overfitting. It can be used with complex and large datasets; however, its main drawback is its slowness, as training takes a long time, prediction is slower compared to several other algorithms, and it consumes a large amount of memory to store the trees [12].

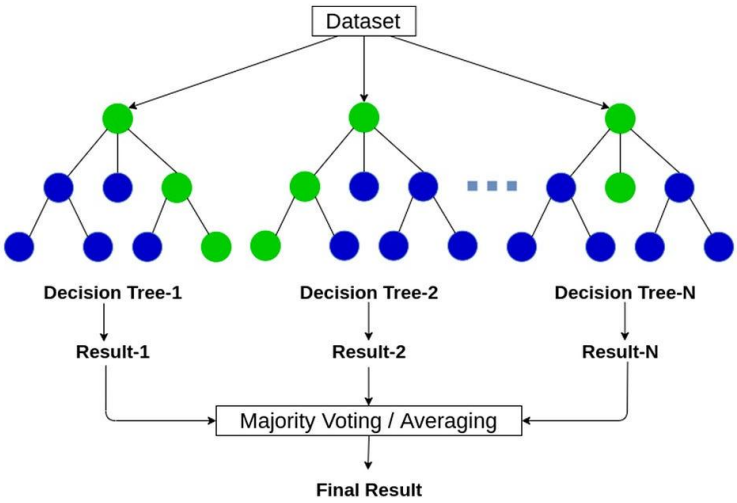

Fig 5. Format of random forest

Random forest in sentiment analysis：

Random Forest uses the same methodology as Decision Trees described earlier, but in a broader way. It relies on voting to determine the classification of an instance, which improves accuracy compared to a single decision tree and helps overcome its limitations, such as sensitivity to changes in the data.

In the study by AL. Yassine et al.[13], a hybrid model combining Random Forest and SVM was proposed to achieve higher accuracy by integrating the strengths of both methods. The Random Forest component builds a large number of decision trees using randomly selected features, and the final class of a new instance is predicted through voting among these trees. The resulting model, called RFSVM , achieved improved precision.

	precision	recall	F-measure
Resulting	83.4%	83.4%	83.4%

The study demonstrated the accuracy and strength of both algori

thms, and showed how combining them through hybridization can produce a more accurate and more powerful model.

2.1.4 Linear regression (LR) :

Linear regression is one of the simplest and most important methods in statistical learning and machine learning. It aims to model the relationship between a dependent variable and one or more independent variables (features), assuming that this relationship can be approximated by a straight line. The main idea is to find the best-fitting line that passes close to the data points, minimizing the difference between the actual values and the predicted values. This difference is known as the error or residual. As can be seen in [fig 6.]

Linear regression is widely used in practical applications such as price prediction, demand estimation, and scientific data analysis. In its simple form (Simple Linear Regression), only one independent variable is used, while in the multiple form (Multiple Linear Regression), several variables are used to explain the dependent variable. The coefficients are estimated to reflect the strength and influence of each independent variable on the final outcome, which helps in better understanding the relationships within the data.

One of its main advantages is that it is easy to understand and interpret, and computationally efficient compared to more complex methods. However, it assumes a linear relationship between variables, which may not always hold true in real-world data. Therefore, its accuracy can be limited in more complex scenarios [14].

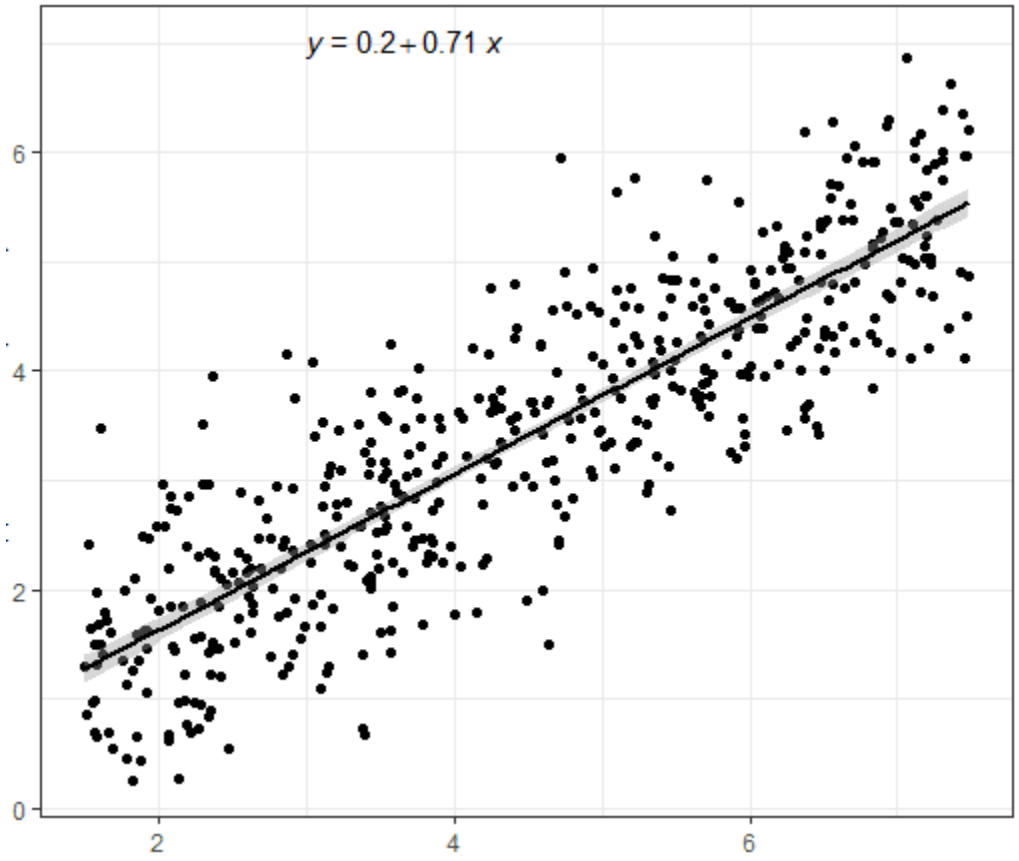

Fig 6. Format of linear regression

Liear regession in sentiment analysis：

Linear regression is used to find a mathematical relationship between inputs and outputs. In sentiment analysis, text is converted into a numerical representation, where each piece of text becomes a vector of numbers. Linear regression then uses these values to calculate a score that represents sentiment, such as values between 0 and 1, or in some cases between -1 and +1 (negative, neutral, or positive).

In a study by K. Nikhil et al.[15], different algorithms were compared. When no preprocessing was applied (i.e., the text was used almost as-is), linear regression performed the best by a small margin of about 1.38%. This is because the model is simple and works well when there are no complex transformations in the data.

2.1.4 Adaboost

AdaBoost (Adaptive Boosting) is a powerful ensemble learning algorithm in machine learning that enhances the performance of weak learners by combining them into a strong classifier. It works by training a series of simple models usually decision stumps where each new model pays more attention to the data points misclassified by

previous models. During training, each data point is assigned a weight, which is adjusted iteratively so that misclassified samples receive higher weights, encouraging subsequent models to focus on them. The final model combines all weak learners through a weighted voting system, where each learner's influence depends on its accuracy. This adaptive approach helps AdaBoost improve predictions and reduce errors over time. As demonstrated in [fig 7.]

AdaBoost's strengths include transforming weak learners into a highly accurate model without requiring complex individual components. It is especially effective for classification tasks and has been widely applied in areas like face detection, text classification, and sentiment analysis. Additionally, it is relatively easy to implement and requires less parameter tuning compared to some advanced models. By emphasizing the most informative data points, AdaBoost often achieves better generalization performance. However, this focus can also be a drawback, as AdaBoost is sensitive to noisy data and outliers; misclassified noisy points gain higher importance and can negatively impact the model's performance.

Despite these limitations, AdaBoost remains a foundational ensemble learning algorithm and has inspired many modern boosting methods such as Gradient Boosting and XGBoost. Its simple concept and strong theoretical basis make it valuable for both research and practical use. In many real-world applications, AdaBoost delivers competitive results, especially when paired with proper preprocessing and well-chosen base learners. As ML advances, AdaBoost continues to be an important step toward understanding more sophisticated ensemble techniques and building robust predictive models [16] .

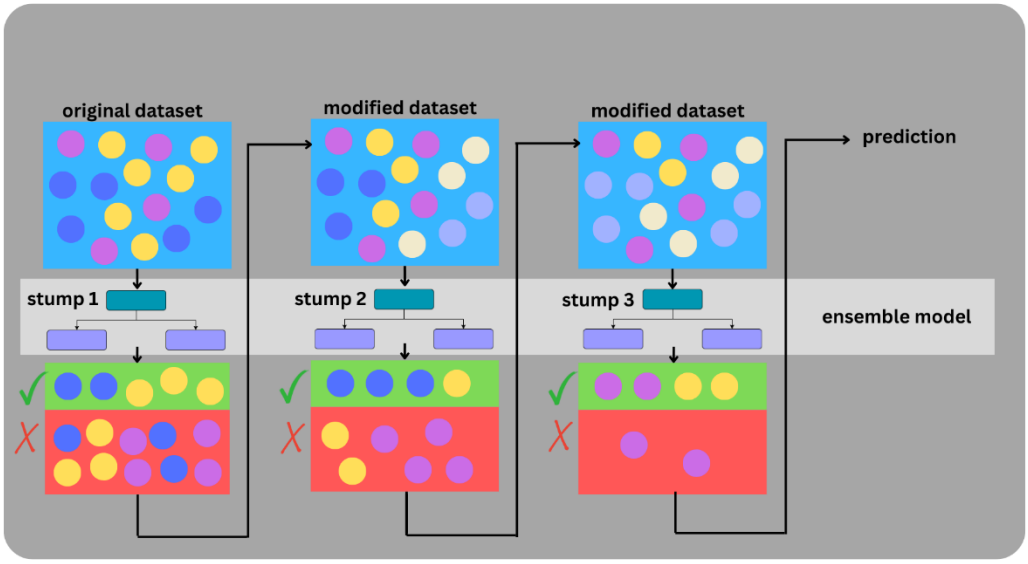

Fig 7. Format of Adaboost

Adaboost in sentiment analysis：

AdaBoost (Adaptive Boosting) is widely used in sentiment analysis to boost classification performance by combining multiple weak learners into a strong predictive model. In this setting, textual data like reviews or tweets are first transformed into numerical features using methods such as Bag-of-Words (BoW) or TF-IDF. These features then train a sequence of simple classifiers, typically decision stumps. AdaBoost iteratively adjusts the weights of training samples, placing greater emphasis on misclassified texts, which helps the model focus on challenging or ambiguous sentiment cases. The final sentiment prediction positive, negative, or neutral is made by a weighted combination of all classifiers, enhancing overall accuracy.

A key advantage of AdaBoost in sentiment analysis is its ability to improve results without relying on complex models. It handles structured textual features effectively and can highlight important sentiment-related words by concentrating on difficult samples. However, AdaBoost is sensitive to noisy or mislabeled data, which are common in real-world text; such samples receive higher weights and can adversely impact performance. Despite this drawback, when paired with appropriate preprocessing and feature engineering, AdaBoost remains a robust and practical method for tasks like opinion mining and social media analysis.

The study by M. Vladimir et al.[17] enhanced AdaBoost's performance in sentiment classification by integrating meta-heuristic optimization. Specifically, the AdaBoost classifier was fine-tuned using a modified Particle Swarm Optimization (PSO) algorithm. This optimized model, named AB-AGbPSO, outperformed other AdaBoost models tuned with different advanced optimization techniques, achieving a highest accuracy of 87.25%.

2.1.5 XGboost :

As we see in [fig 8.], XGBoost (Extreme Gradient Boosting) is one of the most powerful and widely used modern machine learning algorithms for classification and regression tasks, especially in fields such as data analysis, prediction, and sentiment analysis. XGBoost is based on the Gradient Boosting principle, where a strong model is built by combining a large number of weak learners, typically small decision trees. These trees are trained sequentially to correct the errors made by previous ones, which continuously improves the model with each iteration. What makes XGBoost stand out is the set of mathematical and technical optimizations it introduces, making it faster, more accurate, and more stable compared to traditional boosting methods.

One of the key features of XGBoost is the use of regularization techniques that help reduce the problem of overfitting, thereby improving the model's ability to generalize to unseen data. It is also highly efficient when working with large and complex datasets, and it benefits from parallel processing and optimized computations that reduce unnecessary calculations. In sentiment analysis applications, text data is first converted into numerical representations such as TF-IDF or embeddings, and then the model is trained to classify texts into categories such as positive, negative, or neutral. XGBoost often outperforms algorithms like Naïve Bayes and Decision Trees due to its ability to capture complex relationships between features.

Despite its strengths, XGBoost has some challenges, particularly in tuning hyperparameters, which often requires time and multiple experiments to achieve optimal performance. Additionally, its results are less interpretable compared to simpler models such as linear regression or a single decision tree. However, XGBoost remains a fundamental choice in both academic competitions and real-world applications because of its high accuracy, flexibility, and ability to handle different types of data,

making it an ideal algorithm for achieving strong performance in predictive and classification models [18] .

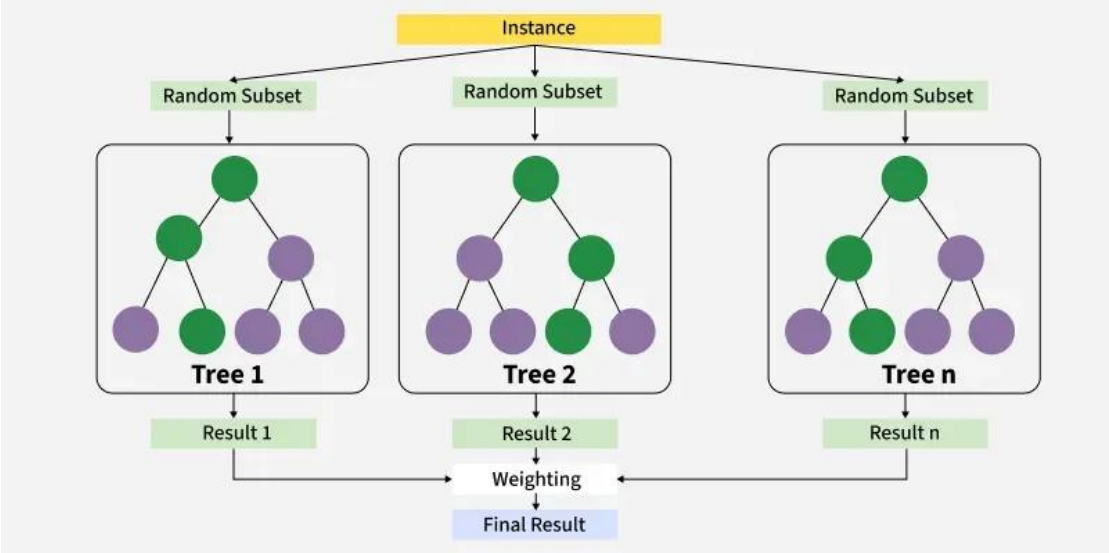

Fig 8. Format of XGboost

XGboost in sentiment analysis：

XGBoost is a highly effective ML algorithm widely used in sentiment analysis for its accuracy and ability to handle complex datasets. It builds a series of weak learners, usually decision trees, where each tree is trained to correct the errors of the previous ones, progressively enhancing the model's overall performance. In sentiment analysis, text data is first transformed into numerical features using techniques like TF-IDF or word embeddings, and then XGBoost classifies the text into categories such as positive, negative, or neutral.

A key advantage of XGBoost is its capacity to capture complex feature relationships, often outperforming traditional algorithms like Naïve Bayes and Decision Trees. It also employs regularization methods that help prevent overfitting, improving the model's reliability on new, unseen data. Optimized for speed and efficiency, XGBoost effectively manages large-scale datasets commonly encountered in social media and review-based sentiment analysis. While it requires careful hyperparameter tuning, XGBoost remains one of the top choices for achieving high-quality sentiment classification results.

The study [19] compared the performance of AdaBoost and XGBoost on IMDb movie reviews categorized into positive and negative classes. The primary objective was to assess how well each algorithm performed in sentiment classification.

The results demonstrated that XGBoost outperformed AdaBoost across all evaluation metrics. It achieved an accuracy of 85%, compared to AdaBoost's 77%. Furthermore, XGBoost also showed superior performance in other key metrics such as precision, recall, and F1-score, indicating its overall advantage for this sentiment analysis task.

2.2 Deep learning in sentiment analysis :

Deep learning in sentiment analysis involves using advanced neural network models to automatically detect and interpret emotions and opinions expressed in text. Unlike traditional machine learning methods, DL models learn complex patterns and contextual relationships directly from raw data, without the need for extensive manual feature engineering. Common architectures include Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and Transformers, which excel at processing sequential and contextual information in language.

Deep learning has greatly enhanced sentiment analysis accuracy, particularly with large and complex datasets like social media posts, product reviews, and customer feedback. By leveraging Natural Language Processing techniques, these models can capture subtle nuances such as sarcasm, tone, and context more effectively than traditional methods. Consequently, DL is widely applied in areas like brand monitoring, opinion mining, and automated customer support systems

2.2.1 Convolutional Neural Network (CNN) :

Convolutional Neural Networks (CNNs), initially developed for computer vision, have proven highly effective in Natural Language Processing (NLP) tasks as well. In NLP, CNNs automatically extract local and position-invariant features from text data. Instead of processing pixels, the input consists of word embeddings arranged in a matrix representing a sentence or document. Convolutional filters slide over this matrix to capture meaningful n-gram patterns such as phrases or short expressions that carry important semantic information. As presented in [fig 9.]

A key advantage of CNNs in NLP is their ability to capture local dependencies without manual feature engineering. For example, in sentiment analysis, a filter might learn to detect phrases like "not good" or "very impressive," which strongly indicate sentiment. Following convolution, a pooling layer commonly max-pooling is applied

to reduce dimensionality and retain the most salient features, enhancing the model's robustness to variations in sentence structure and length.

CNN-based models are widely used in text classification, sentiment analysis, and sentence modeling. While they are less effective at capturing long-range dependencies compared to recurrent or transformer-based models, CNNs offer computational efficiency and strong performance when local context suffices. Consequently, CNNs remain a robust baseline and are often combined with other architectures to boost overall NLP performance [20].

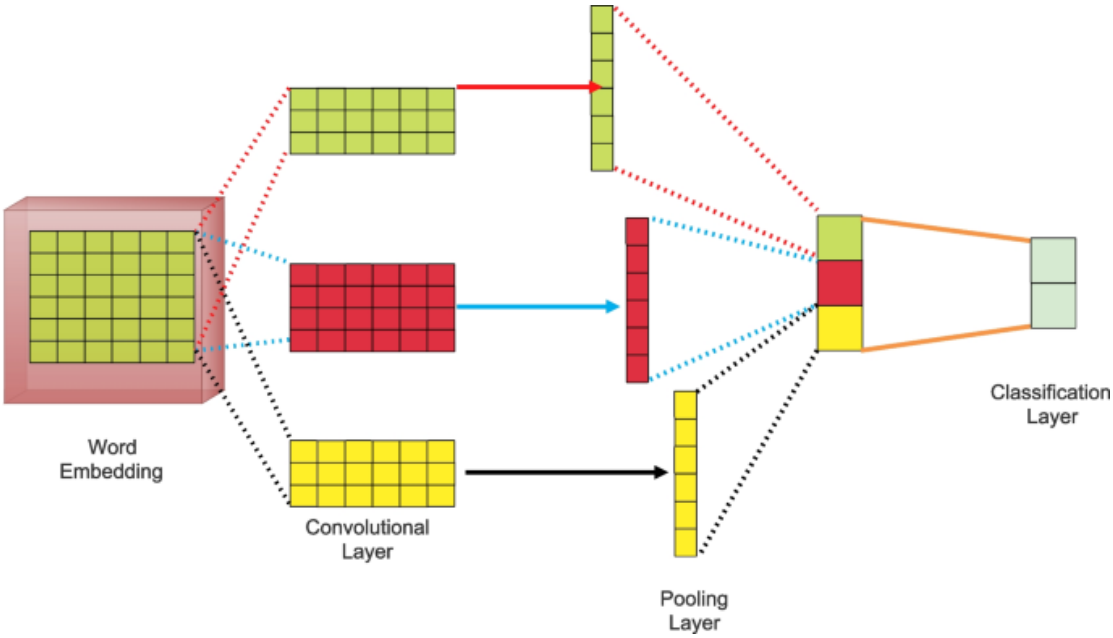

Fig 9. Format of CNN

Convolutional Neural Network in sentiment analysis

Convolutional Neural Networks (CNNs) are widely used in sentiment analysis tasks in Natural Language Processing due to their ability to automatically extract meaningful features from text. In this context, sentences are first transformed into word embeddings, forming a matrix representation. CNNs then apply convolutional filters over this matrix to detect local patterns such as key phrases or n-grams that carry emotional or opinion-related information, for example "absolutely amazing" or "not worth it".

After feature extraction, pooling layers typically max pooling are used to retain the most important signals while reducing dimensionality. This helps the model focus on the strongest sentiment indicators regardless of their position in the sentence. CNN-

based sentiment analysis models are effective, especially for short to medium-length texts, and are valued for their speed and strong performance compared to more complex sequence models.

In the study Z. Qiannan et al.[21], CNN is used to extract local features from text, where filters slide over a matrix of words to detect patterns such as phrases and sentiment-bearing expressions. This capability makes CNN highly effective in capturing important words or phrases that convey emotional meaning within the text.

2.2.2 Long short term memory (Lstm)

Long Short-Term Memory networks (LSTMs) are a specialized type of Recurrent Neural Network (RNN) widely used in Natural Language Processing (NLP) to handle sequential data. Unlike traditional RNNs, LSTMs overcome the vanishing gradient problem, enabling them to capture long-range dependencies in text. This capability makes them particularly effective at understanding context in sentences where the meaning of a word depends on earlier words in the sequence.

In NLP tasks, LSTMs process text word by word while maintaining a memory cell that selectively stores or forgets information over time. This gating mechanism consisting of input, forget, and output gates allows the model to decide which information is relevant for prediction. Consequently, LSTMs are better at understanding complex linguistic structures such as negation or contextual sentiment compared to simpler models. As indicated in [fig 10.]

LSTMs are commonly applied in tasks like sentiment analysis, machine translation, and text generation. For example, in sentiment analysis, they can capture how sentiment shifts across a sentence or paragraph, even when key words are separated by long distances. Although transformer-based models have become dominant recently, LSTMs remain valuable due to their strong sequential modeling abilities and relatively lower computational demands [22].

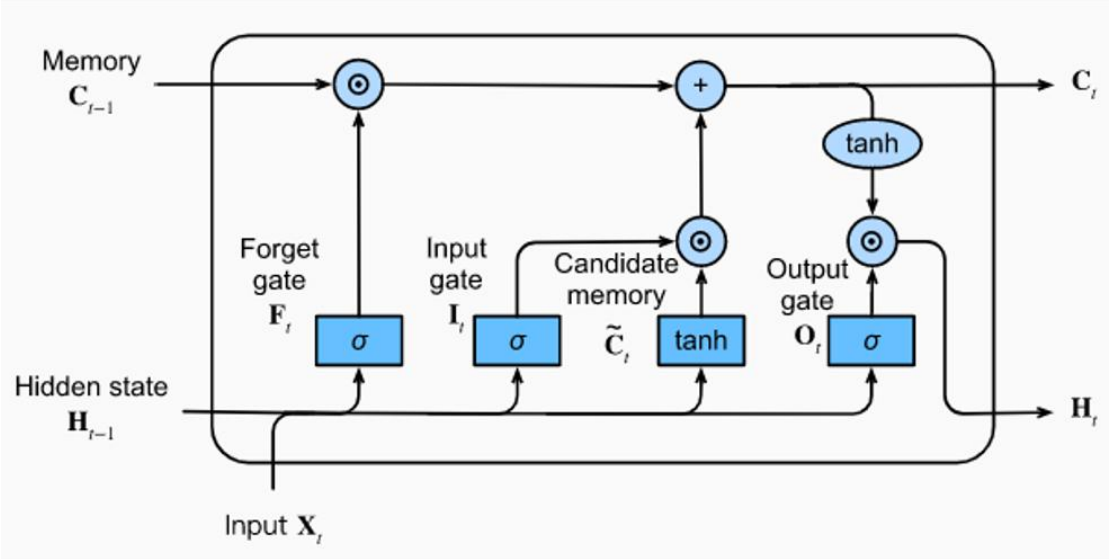

Fig 10. Format of LSTM

Long short term memory in sentiment analysis

Long Short-Term Memory networks (LSTMs) are widely used in sentiment analysis within Natural Language Processing because of their effectiveness in modeling sequential text data. Understanding word order is crucial in sentiment analysis, as sentiment often depends on context spread throughout a sentence. LSTMs process text step by step, maintaining a memory of previous words, which enables them to capture dependencies such as negation or intensity shifts for example, distinguishing between "not good" and "very good."

Thanks to their gating mechanisms, LSTMs can selectively remember or forget information, allowing the model to focus on the most relevant parts of a sentence for predicting sentiment. This capability makes them especially useful for handling longer texts where sentiment is expressed across multiple phrases. Consequently, LSTM-based models often achieve strong performance in sentiment classification tasks, particularly when context and word order play a significant role in meaning.

In the study Dr. G. S. N. Murthy et al.[23], an approach for sentiment classification based on LSTM is proposed using a large amount of data collected from multiple sources such as Facebook and Amazon, which are considered among the most important types of textual data for sentiment analysis. The study shows that deep learning methods like LSTM achieve better performance in sentiment classification, reaching an accuracy of up to 85% when larger amounts of training data are available.

2.2.3 Gated Recurrent Units (GRU)

Gated Recurrent Units (GRUs) are a type of recurrent neural network widely used in Natural Language Processing (NLP) for modeling sequential data. Introduced as a simpler alternative to LSTMs, GRUs capture long-range dependencies in text while using fewer parameters. They process sentences word by word, maintaining a hidden state that carries contextual information across the sequence, which is crucial for understanding language meaning.

Unlike LSTMs, GRUs merge the input and forget gates into a single update gate and include a reset gate that controls how much past information to ignore. This streamlined design enables faster training while still effectively capturing important dependencies in text. As a result, GRUs are efficient for handling large datasets in NLP tasks without significant loss of performance. As depicted in [fig 11.]

GRUs are commonly applied in sentiment analysis, machine translation, and text classification. In sentiment analysis, they help track how sentiment develops throughout a sentence by retaining relevant context and filtering out less important details. While transformer-based architectures often achieve higher accuracy, GRUs remain a strong option when computational efficiency and architectural simplicity are priorities [24]

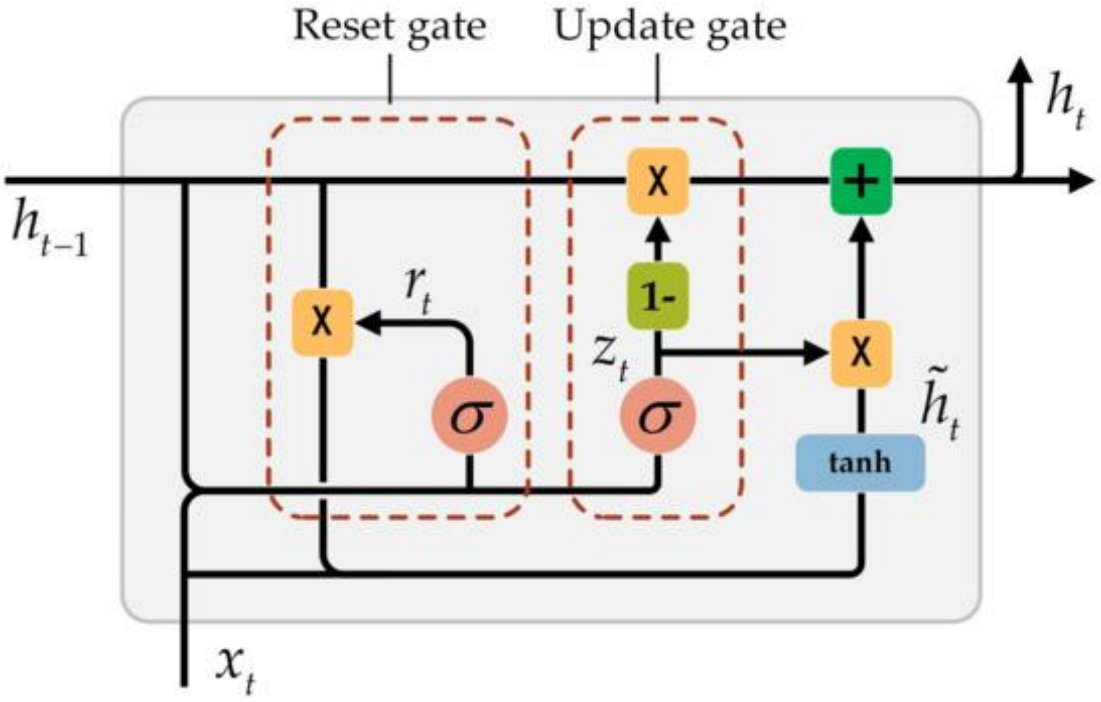

Fig 11. Format of GRU

Gated Recurrent Units in sentiment analysis

Gated Recurrent Units (GRUs) are widely used in sentiment analysis within Natural Language Processing because they effectively model sequential dependencies in text. Sentiment often depends on word order and contextual relationships such as negation or emphasis. GRUs process text step by step, maintaining a hidden state that carries relevant contextual information, enabling the model to track how sentiment develops throughout a sentence.

In sentiment classification, GRUs leverage their update and reset gates to decide which information to retain or discard, allowing them to focus on the most important sentiment-bearing words. This ability helps handle cases where sentiment spans longer phrases or relies on earlier context. Thanks to their simpler structure compared to LSTMs, GRUs typically train faster while maintaining strong performance in many sentiment analysis tasks.

In the paper A.WAQAR et al.[25], GRUs address the issue of information loss common in traditional models by employing gating mechanisms that regulate which information to retain or discard. This capability enables the model to effectively handle long and complex texts, improving the accuracy of aspect-level sentiment analysis.

2.3 Transformers

Transformers are a deep learning architecture specifically designed to process sequential data, particularly in language understanding tasks. Introduced in the paper "Attention Is All You Need," Transformers leverage a self-attention mechanism that enables the model to assess the importance of different words in a sentence regardless of their position. This capability allows Transformers to capture context and relationships between words more effectively than earlier models like recurrent neural networks. As a result, Transformers have become a foundational technology in modern natural language processing.

In NLP, Transformers have transformed tasks such as translation, text summarization, and sentiment analysis. Prominent models like BERT and GPT are built on Transformer architecture and achieve state-of-the-art performance by training on large-scale datasets. Their ability to manage long-range dependencies and parallelize computations makes them both powerful and efficient, driving widespread adoption in both research and industry.

2.3.1 BERT (Bidirectional Encoder Representations from Transformers) :

The Transformer architecture marks a significant breakthrough in Natural Language Processing by relying solely on attention mechanisms instead of sequential processing. It employs self-attention to examine relationships between all words in a sentence simultaneously, efficiently capturing both local and global context. This parallel processing capability enhances performance and scalability, making Transformers highly effective across diverse language understanding tasks.

BERT (Bidirectional Encoder Representations from Transformers) is a model built on the Transformer encoder architecture, designed to understand context bidirectionally. Unlike traditional models that read text in one direction, BERT considers both the left and right context of each word simultaneously, enabling a deeper comprehension of word meanings, especially when context heavily influences interpretation.

Pre-trained on large text corpora through objectives like masked language modeling and next sentence prediction, BERT learns rich linguistic representations.

These pre-trained models can then be fine-tuned for specific NLP tasks such as sentiment analysis, question answering, and text classification. By harnessing the Transformer's power, BERT delivers high performance and versatility across various language applications [26].

BERT in sentiment analysis

BERT (Bidirectional Encoder Representations from Transformers) excels in sentiment analysis because it captures the full context of words by processing text bidirectionally. This means it understands how each word relates to both preceding and succeeding words an essential feature for accurately interpreting sentiment. For example, phrases like "not bad" or "hardly useful" require contextual awareness that BERT models more effectively than traditional methods.

In sentiment classification, BERT is typically fine-tuned on labeled datasets, learning to associate textual patterns with sentiment labels such as positive or negative. Its deep contextual representations enable it to detect subtle emotional cues, sarcasm, and complex sentence structures. Consequently, BERT-based models often achieve high accuracy in sentiment analysis across diverse text types, including reviews, social media posts, and customer feedback.

H. Mickel et al.[27] introduces a model that leverages the pre-trained BERT architecture, fine-tuned specifically for sentence pair classification tasks, to perform aspect-based sentiment analysis effectively. A key advantage of this model is its ability to handle both in-domain and out-of-domain data by learning the relationships between aspects and corresponding texts using automatically generated labeled data. Additionally, the model employs a combined approach that integrates aspect detection and sentiment classification into a single unified framework. Experimental results show that this method enhances performance and surpasses previous state-of-the-art techniques in aspect-based sentiment analysis.

2.3.2 ARABERT (ARABIC BERT)

The Transformer architecture has significantly advanced Natural Language Processing by introducing self-attention mechanisms that allow models to understand relationships between words in a sentence regardless of their position. This approach enables parallel processing of text and captures both short- and long-range

dependencies effectively. As a result, Transformers are particularly well-suited for complex language understanding tasks where context plays a critical role.

ARABERT is a language model built on the Transformer encoder architecture, specifically designed for processing Arabic text. It takes into account the linguistic characteristics of Arabic, such as rich morphology, complex word formations, and dialectal variations. By leveraging the bidirectional nature of the Transformer, ARABERT can better understand context within Arabic sentences, improving performance in tasks that require deep semantic understanding.

ARABERT is pre-trained on lar Aspect-Based Sentiment Analysis Using BERT ge-scale Arabic corpora, allowing it to learn meaningful representations of words and phrases in different contexts. These representations can then be fine-tuned for specific tasks such as sentiment analysis, text classification, and question answering in Arabic. By utilizing the Transformer architecture, ARABERT achieves strong performance while addressing the unique challenges associated with the Arabic language [28].

ARABERT in sentiment analysis

ARABERT, a Transformer-based model specifically designed for Arabic text, excels in sentiment analysis by effectively capturing the linguistic complexity of Arabic. The Arabic language poses challenges such as rich morphology, diverse dialects, and flexible word order, all requiring deep contextual understanding. Leveraging the bidirectional attention mechanism of the Transformer, ARABERT can interpret sentiment with greater accuracy, even when conveyed through subtle phrasing or context-dependent meanings.

In sentiment classification tasks, ARABERT is typically fine-tuned on labeled Arabic datasets, enabling it to learn how various expressions correspond to positive, negative, or neutral sentiments. Its pre-trained knowledge allows it to recognize patterns in both Modern Standard Arabic and, to some extent, dialectal variations. This makes ARABERT particularly effective for analyzing social media posts, reviews, and user feedback, where language use varies widely but still carries clear emotional intent.

A. Wissam et al.[29] shows that the model achieved highly advanced results in several different tasks, such as sentiment analysis. It also indicates that it is 300 MB smaller

compared to the multilingual BERT model, which makes it more efficient in terms of storage and usage.

2.3.3 DZIRIBERT

The Transformer architecture has been pivotal in advancing Natural Language Processing by introducing self-attention mechanisms that enable models to capture relationships between all words in a sentence simultaneously. This design allows for parallel processing and a richer understanding of context compared to traditional sequential models, making Transformers highly effective for tasks requiring nuanced language comprehension and contextual awareness.

DZIRIBERT is a Transformer-based language model specifically developed for the Algerian Arabic dialect (Darija). It is tailored to address the unique linguistic features of Algerian speech, which often blends Arabic, French, and other influences. By leveraging the Transformer's bidirectional context modeling, DziriBERT can better understand informal expressions, code-switching, and language variations common in Algerian text, particularly in social media and everyday communication.

Pre-trained on large corpora of Algerian dialect data, DZIRIBERT learns meaningful representations of words and phrases in their natural context. These representations can be fine-tuned for various NLP tasks such as sentiment analysis, text classification, and dialect identification. By harnessing the strengths of the Transformer architecture, DZIRIBERT offers enhanced performance in applications that demand accurate understanding of Algerian Arabic.

DZIRIBERT sentiment analysis

DZIRIBERT is particularly effective in sentiment analysis for Algerian Arabic (Darija) because it is designed to handle the informal and highly variable nature of the dialect. In real-world data such as social media posts, users often mix Arabic, French, and slang expressions, making sentiment detection more challenging. By leveraging the Transformer's bidirectional attention mechanism, DZIRIBERT can capture contextual meaning more accurately, even when sentiment is expressed implicitly or through code-switching.

In sentiment classification tasks, DZIRIBERT is typically fine-tuned on labeled datasets that reflect the diversity of Algerian language usage. This allows the model to

learn how different phrases and expressions correspond to positive, negative, or neutral sentiments. Its ability to understand local expressions and contextual nuances makes it especially valuable for analyzing user opinions, reviews, and online discussions within the Algerian context.

A. Amine et al.[30] demonstrates that DZIRIBERT achieves strong performance across various Natural Language Processing tasks, even when trained on relatively limited data. This underscores the model's effectiveness in handling low-resource languages like the Algerian dialect, which are often underrepresented in NLP research. The authors also highlight the significance of making the pre-trained model publicly available, along with fine-tuned versions for applications such as sentiment analysis, emotion detection, and topic classification, to encourage further research and practical use.

Chapter 3 System Architecture and Model Development

3.1. Preprocessing text cleaning (overview) :

Text preprocessing is an important step in Natural Language Processing (NLP). However, with the emergence of modern deep learning approaches particularly transformer-based models such as BERT the role of preprocessing has evolved. Traditional text cleaning techniques are no longer applied extensively, as many of these processes are handled implicitly by pretrained models.

Text preprocessing is the process of converting raw textual data into a clean and structured format that is ready for analysis. Unlike numerical data, text often includes irregularities like spelling variations, punctuation marks, emojis, and irrelevant symbols. Cleaning and standardizing this data is crucial to ensure consistency and enhance the accuracy of machine learning models.

To clean text is an important step, Effective preprocessing offers several benefits :

- Improved Model Accuracy ;
- Reduced Dimensionality ;
- Faster Processing ;
- Better Generalization .

3.1.1. Common Text Cleaning Techniques :

- **Lowercasing**

Converting all text to lowercase ensures uniformity in the dataset. This way, words like “Apple” and “apple” are treated as the same token, reducing redundancy and helping the model better recognize patterns.

- **Removing Punctuation**

Punctuation marks usually do not add meaningful information in many NLP tasks. Removing punctuation simplifies the text and helps reduce noise, making the data cleaner for analysis.

- **Removing Numbers**

Depending on the task, numbers may or may not be relevant. For sentiment analysis, numbers are often removed unless they carry specific significance, as they may not contribute to understanding sentiment.

- **Removing Extra Whitespace**

Text data often contains unnecessary spaces, tabs, or line breaks. Cleaning these extra whitespaces ensures consistency and prevents issues in tokenization and further processing.

3.1.2. Core Text Processing Techniques :

Tokenization is the process of breaking text into smaller, meaningful units called tokens. These tokens can be individual words, subwords, or even entire sentences, depending on the specific task. For example, the sentence “This is a sample sentence” can be tokenized into the words “This,” “is,” “a,” “sample,” and “sentence.” Tokenization is a crucial step because most NLP models operate on these smaller units rather than raw text.

To illustrate, consider a sentence in the Algerian dialect: “راك فاهم وش نقول ؟” After tokenization, it might be split into: “راك”، “فاهم”، “وش”، “نقول”. This enables the model to process each component of the sentence separately, facilitating a better understanding of its structure and meaning

3.1.3. Advanced Cleaning and Normalization

Handling special elements is a crucial part of text preprocessing. Text gathered from the internet often includes URLs, HTML tags, emojis, and various special characters. Typically, URLs and HTML tags are removed as they usually don't add meaningful information. However, emojis can convey strong emotional cues for example, the phrase “هذا الفيلم هايل ” clearly expresses positive sentiment, and removing the emoji might reduce this clarity. In some cases, emojis are converted into descriptive words rather than discarded. Special characters like @, #, and $ frequently appear in social media texts, such as “#روعة كان اليوم ماتش,” and how they are treated depends on the specific application.

Normalization aims to make text consistent and standardized. Common normalization steps include expanding contractions, for example, converting “don’t” to

"do not" or "I'm" to "I am," enhancing clarity across the dataset. In dialectal contexts like Algerian Arabic, normalization involves unifying different spellings of the same word for instance, "راكْ" ،"راك" ،or "rak" in Latin script. Spelling correction is also important, fixing errors such as "recieve" to "receive." Dialectal variants like "بزاف," "bzaf," or "bezzaf" are standardized through normalization. Another useful technique addresses repeated characters that indicate emphasis, converting exaggerated expressions like "زووووين" or "soooo good" into "زوين" and "so good" respectively.

3.1.4. Data Quality and Feature Preparation

Dealing with imbalanced and noisy data is a crucial step in real-world text applications. Datasets often have an uneven distribution of classes for example, more positive reviews than negative ones which can bias model training. Additionally, text data may contain irrelevant content, duplicates, or incorrect labels that degrade model quality. Preprocessing addresses these challenges by removing duplicate entries, filtering out unrelated or noisy text, and cleaning the dataset as thoroughly as possible. For instance, repeated expressions like "هايل هايل هايل" can be reduced or managed based on the specific goals of the analysis.

Once preprocessing is complete, the text can be transformed into numerical representations using techniques such as Bag of Words, TF-IDF, or word embeddings. However, in this work, contextual embeddings generated by pretrained models are used instead.

3.1.5. Challenges and Best Practices in Text Preprocessing

Despite its critical role, text preprocessing faces several challenges. One major issue is language dependency-techniques effective for English may not directly apply to Arabic or Algerian dialects. Context sensitivity also poses difficulties, as removing or altering certain words can change the intended meaning of a sentence. Additionally, domain-specific requirements matter; texts from medical, legal, or social media fields often need tailored preprocessing strategies. There is an inherent trade-off between cleaning the text and preserving valuable information, since over-cleaning risks losing important details.

To address these challenges effectively, best practices should be followed. It is essential to thoroughly understand the dataset before applying preprocessing and to customize techniques according to the specific task. Care must be taken not to remove critical words like negations, and different approaches should be tested and evaluated for their impact on model performance. Striking the right balance between simplicity and thoroughness is key to developing a robust and effective preprocessing pipeline

3.2. preprocessing feature extraction :

In Natural Language Processing (NLP), preprocessing and feature extraction are two closely interconnected stages that convert raw text into meaningful numerical representations suitable for ML models. Preprocessing involves cleaning and organizing the text to remove noise and inconsistencies, while feature extraction transforms this cleaned text into numerical vectors. These stages are highly dependent on each other, as inadequate preprocessing can produce noisy or irrelevant features, which in turn negatively impact the performance of the resulting models.

3.2.1. The Relationship Between Preprocessing and Feature Extraction

Before extracting features, text must be standardized and cleaned because raw text is often messy and inconsistent, making it challenging for algorithms to learn effectively. For example, the sentence “This movie is AMAZING!!!” can be simplified through preprocessing to “movie amazing.” Similarly, in a more local context using Algerian dialect, a sentence like “هاد الفيلم راهو هاااايل بزاف!!! 😍” can be cleaned and normalized to “الفيلم هايل بزاف.” Once the text reaches this clean form, feature extraction techniques convert it into numerical vectors that capture its meaning.

This process functions as a continuous pipeline where raw text is gradually transformed into a format suitable for machine learning. It starts with raw text, goes through preprocessing to yield clean text, and then proceeds to feature extraction, where the final numerical representations are generated and passed to the model for training or prediction

3.2.2. Preprocessing as the Foundation

Preprocessing acts as the foundation for all subsequent مراحل (stages). It includes several key steps such as converting text to lowercase, removing punctuation and special characters, tokenization, and normalization. Additionally, it may involve removing stopwords and applying techniques like stemming or lemmatization.

For instance, a sentence in Algerian dialect like "أناااااا راني نحوس على فيلم مليح!!!" can be transformed into a cleaner version such as "راني نحوس فيلم مليح". This process eliminates unnecessary repetitions, simplifies the sentence structure, and retains only the meaningful components. Such transformations reduce noise and improve data consistency, which are essential prerequisites before proceeding to feature extraction.

3.2.3. Understanding Feature Extraction

Feature extraction is the process of transforming text into numerical vectors, enabling machines to interpret and work with textual data. Since computers cannot directly understand words, these vectors mathematically represent the meaning, structure, or significance of the words within the text.

For example, a simple Algerian sentence like "القعدة كانت زينة بزاف" can be converted into a vector representation using various methods such as Bag of Words, TF-IDF, or word embeddings. The effectiveness and quality of this representation largely depend on the thoroughness and accuracy of the preceding preprocessing stage

3.2.4. Common Feature Extraction Techniques

Here's a concise explanation of common feature extraction techniques in NLP, illustrated with examples from Algerian dialect:

- **Bag of Words (BoW) :**

BoW represents text based on word frequency. A vocabulary is built from all unique words in the dataset, and each sentence is encoded by counting how often each word appears. For example, the sentences "نحب القهوة" and "نحب الشاي" produce a vocabulary: "الشاي"، "القهوة"، "نحب". Each sentence is then represented numerically by the presence or absence of these words. While simple and effective for basic tasks, BoW ignores word order and deeper semantic meaning ;

- **N-grams :**

N-grams extend BoW by capturing sequences of words, preserving some context. For instance, the sentence "نحب ناكل كسكسي" can generate bigrams like "نحب ناكل" and "ناكل كسكسي". This helps models understand relationships between neighboring words rather than treating words independently ;

- **TF-IDF (Term Frequency-Inverse Document Frequency) :**

TF-IDF improves on BoW by weighting words according to their importance. Common words like "نحب" get lower weights, while more specific words like "كسكسي" receive higher weights based on their rarity across the dataset. This makes TF-IDF more informative than simple frequency counts ;

- **Word Embeddings :**

These represent words as dense vectors capturing semantic similarity. Words with related meanings, such as "مليح" and "هايل" in Algerian dialect, are positioned close together in the vector space. Embeddings enable models to grasp semantic relationships beyond simple word counts ;

- **Contextual Embeddings :**

The most advanced technique, contextual embeddings capture word meanings depending on context. For example, "بانك" could mean a financial institution or something else based on the sentence, and "ثقيل" could mean "heavy" or "annoying" depending on usage. These models provide nuanced understanding but require significant computational resources.

Each technique offers a trade-off between complexity and representational power, with more advanced methods generally leading to better model performance in complex NLP tasks.

3.2.5. Feature Selection and Challenges

It is important to distinguish between feature extraction and feature selection. Feature extraction involves creating new numerical representations from raw data, transforming text into vectors that capture its properties. In contrast, feature selection focuses on identifying and retaining the most relevant features while removing

redundant or irrelevant ones. This process helps reduce dimensionality, which in turn improves model efficiency and generalization.

Feature extraction, however, presents several challenges. One major issue is high dimensionality, especially with techniques like Bag of Words and n-grams, where the feature space can grow very large. Another challenge is sparsity, as many entries in these representations are often zero, making the data sparse and sometimes difficult to model effectively. Furthermore, traditional feature extraction methods may fail to capture the contextual meaning of words, limiting their effectiveness. While advanced methods such as word and contextual embeddings address this limitation, they demand significant computational resources, which can be a constraint in many practical applications.

3.2.6. Choosing the Right Approach and Best Practices

Selecting the appropriate feature extraction method depends largely on the specific task and the computational resources available. For simpler tasks like sentiment analysis, basic techniques such as Bag of Words or TF-IDF often provide satisfactory results. For more complex tasks, combining n-grams with TF-IDF can enhance performance by capturing additional context. In advanced applications like chatbots or machine translation, word embeddings and contextual models are typically more effective due to their ability to capture semantic nuances.

To achieve optimal results, proper text preprocessing before feature extraction is essential. A practical strategy is to start with simpler methods and progressively adopt more complex techniques as needed. Careful evaluation of model performance and attention to avoiding overfitting by reducing redundant or irrelevant features are also critical. Finally, computational constraints should always be taken into account when selecting a feature extraction approach to ensure efficient and feasible implementation

3.3. Model Architecture

Model architecture in machine learning, especially in Natural Language Processing (NLP), refers to the structured design that dictates how a model processes input data and generates output. It serves as the blueprint outlining the organization of different components, their interactions, and the flow of information within the system. Choosing the right architecture is a critical step in building an NLP system, as it

significantly influences the model's capacity to understand language, capture contextual nuances, and generalize effectively to new data.

At its core, model architecture defines the type of model ranging from simple linear models to complex deep neural networks as well as the number and arrangement of layers. It also specifies how data transitions from the input stage through various transformations to produce the final output. In essence, the architecture describes how the model "thinks" and learns from the data it receives.

3.3.1. Core Components of Model Architecture

Every machine learning model, regardless of its complexity, is composed of core components that work together to process information effectively :

- **Input Layer:**

This serves as the entry point for data. In NLP tasks, the input is not raw text but a processed numerical representation. For example, the Algerian dialect sentence "الفيلم هذا كان هايل بزاف" would first be tokenized and converted into vectors using methods like embeddings or TF-IDF before entering the model ;

- **Hidden Layers:**

After the input layer, data passes through one or more hidden layers responsible for learning patterns and extracting meaningful features. In simpler models, there may be a single transformation step, whereas DL architectures typically stack multiple layers. Each layer applies mathematical transformations that progressively abstract the data. For instance, in sentiment analysis, early layers might identify individual words such as "هايل" or "كارثة," while deeper layers capture more complex patterns like overall sentiment or tone ;

- **Output Layer:**

The final component generates the model's prediction. Depending on the task, this could be a binary label (e.g., positive or negative sentiment), multiple categories, or even a sequence of words for tasks like translation. For example, the sentence "الخدمة اليوم كانت متعبة بزاف" might be classified as expressing negative sentiment, whereas "الرحلة كانت روعة" would be classified as positive.

3.3.2. Evolution of Model Architectures in NLP

Model architectures in NLP have evolved substantially, progressing from simple statistical methods to sophisticated deep learning systems :

- **Traditional Machine Learning Models :**

Early approaches included models like logistic regression, Naive Bayes, and Support Vector Machines. These rely on manually engineered features such as TF IDF vectors. They are straightforward, fast, and effective for many basic tasks, especially with smaller datasets. However, they treat text as a bag of independent features, ignoring word order and contextual information ;

- **Feedforward Neural Networks :**

These models process data in a single forward pass from input to output and can learn non linear relationships. Despite this, they lack the capability to handle sequential data, which is crucial in language understanding ;

- **Recurrent Neural Networks (RNNs)**

RNNs were designed to process sequences by maintaining a hidden state that acts as memory, allowing them to capture word order. For example, in the Algerian sentence "ماشي مليح هذا الفيلم," the position of "ماشي" is essential to grasp the negative sentiment. However, standard RNNs struggle with long sequences due to information fading over time ;

- **Long Short Term Memory Networks (LSTMs) :**

LSTMs enhance RNNs by introducing gating mechanisms that regulate memory retention and forgetting, enabling them to capture long term dependencies effectively. For example, in the sentence "الفيلم بدا مليح بصح في الأخير كان مخيب," LSTMs can remember both the beginning and the end to understand the overall sentiment ;

- **Gated Recurrent Units (GRUs) :**

GRUs are a streamlined variant of LSTMs with fewer parameters, offering computational efficiency while maintaining strong performance. They are suitable when resources are limited ;

- **Convolutional Neural Networks (CNNs)**

Originally developed for image processing, CNNs have been adapted to text by using filters to detect local patterns like phrases or key expressions. For instance, expressions such as "هايل بزاف" or "ما يعجبش" serve as strong sentiment indicators. CNNs

are efficient and effective in tasks like sentence classification but are less capable of capturing long range dependencies.

Each architecture brings unique strengths and trade offs, and the choice depends on the specific requirements of the NLP task at hand.

3.3.3. Transformer Architecture and Modern Advances

The introduction of the transformer architecture revolutionized NLP by moving away from sequential processing used in RNNs and LSTMs. Instead, transformers use a mechanism called self attention, which enables the model to consider all words in a sentence simultaneously and assess their relationships.

This approach allows transformers to capture complex context and dependencies more effectively. For example, in the sentence "الفيلم ما كانش هايل كيما توقعت," the model can understand that "ما كانش" negates the expectation linked to "هايل." Self attention lets each word "attend" to every other word, enhancing the model's ability to grasp nuanced meanings.

Transformers also incorporate embedding layers that convert words into dense vector representations, and positional encoding to inject information about word order. These components enable the model to preserve both semantic meaning and structural information without relying on sequential data processing.

Thanks to their scalability and superior performance, transformer architectures form the foundation of modern NLP systems and advanced language models.

3.3.4. Practical Model Architecture Pipelines

In practice, model architectures are often implemented as pipelines that combine multiple processing steps to handle NLP tasks efficiently.

A simple pipeline might include preprocessing, TF IDF feature extraction, and a logistic regression model. This setup is straightforward, computationally efficient, and well suited for basic classification problems.

More advanced pipelines incorporate deep learning components. For example, a typical DL pipeline may consist of tokenization, an embedding layer, followed by an LSTM or GRU layer, and finally dense layers that generate the output. Such architectures can capture complex patterns and dependencies in the text.

Transformer based pipelines offer even greater power and flexibility. They usually begin with tokenization, followed by embeddings combined with positional encoding, multiple stacked transformer layers, and a final output layer. These pipelines excel in sophisticated applications like machine translation, chatbots, and text generation, delivering state of the art performance.

3.4. Model training

Model training is the phase in machine learning where a model learns patterns from data by adjusting its internal parameters to minimize error. After defining the model architecture and preparing the data, training turns the model into a useful tool. This process involves feeding data into the model, measuring how far its predictions deviate from the true answers using a loss function, and iteratively updating the model's parameters to improve performance.

Central to training is optimization: the model makes predictions, calculates the error (loss), and adjusts parameters to reduce this loss step by step. The goal is to find parameters that minimize the loss function while maintaining good generalization to unseen data.

A key component is the optimizer, the algorithm that updates model weights based on the computed error. The simplest optimizer is Gradient Descent, which moves parameters opposite to the gradient of the loss function. However, it can be slow for large datasets. Variants like Stochastic Gradient Descent (SGD) update weights using one sample at a time, improving speed and scalability. Mini batch Gradient Descent balances stability and efficiency by using small batches of data. More advanced optimizers such as Adam, RMSprop, and Adagrad adapt learning dynamically with Adam being popular for combining momentum and adaptive learning rates, leading to faster convergence and better performance.

The learning rate controls the size of parameter updates. Too high a learning rate risks overshooting the optimal solution, while too low slows down training and can cause the model to get stuck in local minima. Learning rate scheduling adjusts this rate during training to improve convergence.

Training proceeds in epochs and batches. An epoch is one full pass through the entire training dataset, which is often divided into smaller batches for faster and more

stable updates. The number of epochs affects learning: too few can cause underfitting, too many lead to overfitting (memorizing training data rather than generalizing).

The loss function quantifies the difference between predictions and true values. Common loss functions include cross-entropy for classification and mean squared error for regression. The optimizer uses this loss to guide parameter updates.

Regularization techniques help prevent overfitting. Methods like dropout randomly deactivate parts of the network during training, encouraging the model to learn robust features. Other methods, such as L1 and L2 regularization, add penalties to the loss function to discourage overly complex models.

Validation is crucial during training. The dataset is typically split into training and validation sets. The model's performance on the validation set is monitored to detect overfitting or underfitting. Techniques like early stopping halt training once validation performance ceases to improve.

Training deep learning models, especially in NLP, can be computationally intensive, often requiring GPUs or TPUs. The time needed depends on dataset size, model complexity, and hyperparameter choices.

In summary, model training is an iterative optimization process where the model learns by minimizing loss through repeated updates. Optimizers steer learning, learning rates control update magnitude, and epochs and batches define data flow. Careful tuning, regularization, and validation are essential to build models that perform well both on training data and unseen examples

3.5. Overall scheme

In the [Fig 12.] and [fig 13.] illustrate the overall schema and workflow of the proposed program. The first one offers a simplified overview of the system architecture, while the second presents a more detailed view of the internal processes and the interactions between the various components of the application.

These figures show how the system processes input text through multiple stages, starting with text preprocessing and feature extraction, followed by the application of machine learning and deep learning models for sentiment classification. The workflow

ultimately enables the system to determine whether the sentiment is positive or negative and generate the corresponding output.

Additionally, the diagrams clarify the program's general operation and provide a visual representation of the data flow through each analysis stage. A more detailed explanation of the workflow, including the role of each component and the inner workings of the models, will be provided in the next chapter

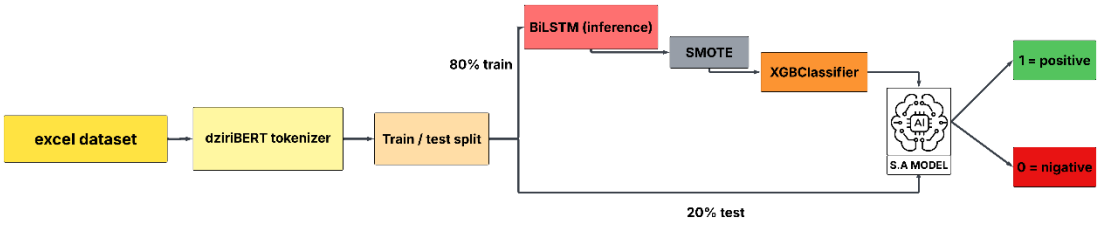

Fig 12. Simple schema

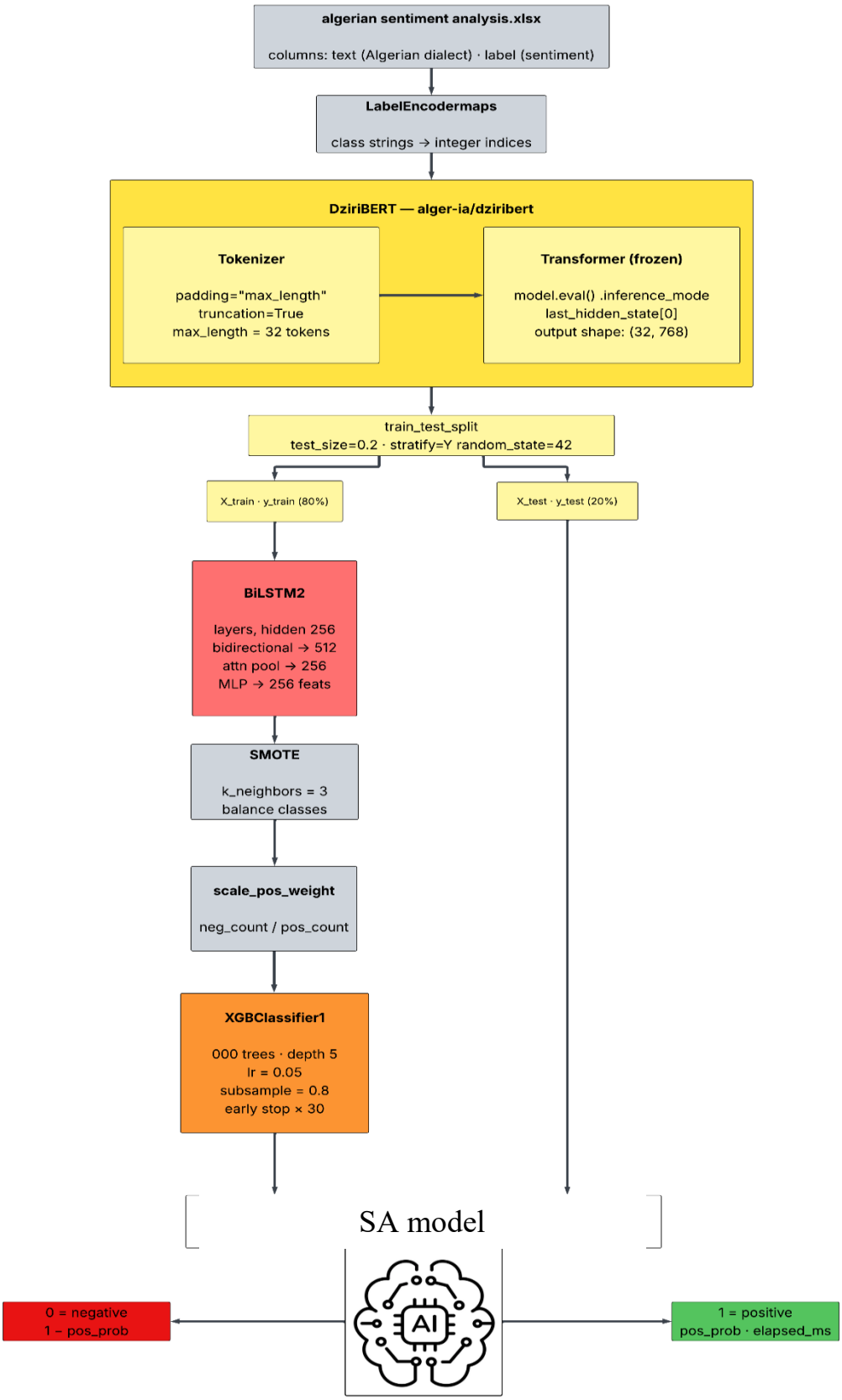

Fig 13. Schema global

Chapter 4 System Deployment, Performance Evaluation, and User Interface

4.1. Machine learning

We experimented with several machine learning models on our dataset to evaluate their performance. The results obtained from each model are presented in the tables below.

To effectively compare the models, we will rely on a set of evaluation metrics and statistics provided by each model. These metrics enable us to assess the efficiency of each model and quantify its ability to correctly classify sentiments. They also facilitate the identification of each model's strengths and weaknesses, allowing us to determine which performs best in sentiment analysis tasks.

The most important evaluation metrics include:

Precision:

Precision measures the accuracy of the model's positive or negative predictions. Specifically, when the model predicts a text as expressing a particular sentiment (e.g., negative), Precision indicates the proportion of those predictions that are actually correct according to the ground truth. For example, if the model classifies several texts as negative, Precision reflects how many of those texts are truly negative in the dataset. This metric is critical because it reflects the reliability of the model's predictions. A higher Precision value signifies fewer false positives and more accurate classification , like we show in Equation 01.

$$Precision = \frac{true\ Positives}{true\ Positives + false\ Positives} \quad \dots\ Equation\ 01$$

Recall:

Recall assesses the model's ability to identify all relevant cases in the dataset. More precisely, among all the truly negative texts in the test set, Recall measures how many the model correctly identifies as negative. The same principle applies to positive texts. Recall is important because it evaluates the model's capacity to minimize false negatives, ensuring that actual cases are not overlooked. A high Recall indicates that

the model detects most relevant instances, even if some classification errors occur, as present in formal Equation 02

$$Recall = \frac{true\ Positives}{true\ Positives + false\ Negatives} \quad \dots Equation\ 02$$

F1-score:

The F1-score provides a harmonic mean of Precision and Recall, offering a single metric that balances the trade-off between these two aspects. Sometimes, a model may achieve high Precision but low Recall, or vice versa. The F1-score addresses this by combining both metrics into one comprehensive value as we see in equation 03:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad \dots Equation\ 03$$

A score closer to 1 indicates a better balance between Precision and Recall, signifying stronger and more consistent model performance.

Accuracy:

Accuracy represents the proportion of all correct predictions made by the model relative to the total number of instances in the test dataset. It measures how many cases positive or negative the model classified correctly. While Accuracy provides an overall performance view and is widely used, it should be interpreted cautiously, especially with imbalanced datasets. A model may achieve high Accuracy by predominantly predicting the majority class, yet perform poorly in identifying minority classes. Therefore, Precision, Recall, and F1-score are also essential to provide a more nuanced and comprehensive evaluation of model performance , like we see in Equation 04

$$Accracy = \frac{correct\ predictions}{all\ predictions} \quad \dots Equation\ 04$$

4.1.1. KNN

As observed from the table, the K-Nearest Neighbors (KNN) model demonstrates varied performance that is generally acceptable for sentiment analysis tasks, exhibiting clear strengths in some areas and notable weaknesses in others.

In terms of Precision, the model shows relatively strong results. It attains a precision score of 78% for the negative class, indicating that the majority of instances predicted as negative were correctly classified. More significantly, it achieves a very high precision of 96% for the positive class, suggesting that when the model predicts a text as positive, it does so with high confidence and accuracy. This reflects a strong capability in correctly identifying positive predictions.

Regarding the F1-score, the model reveals an uneven balance between precision and recall. It records a robust 87% for the negative class, signifying stable and effective identification of negative sentiments. However, this score decreases to 71% for the positive class, indicating comparatively weaker and less consistent performance. While still acceptable in certain contexts, this discrepancy highlights a clear imbalance between the two sentiment classes.

In terms of Accuracy, the model achieves an overall score of approximately 82%, which is considered a good result. This indicates that the model is generally capable of correctly classifying a substantial portion of the dataset, making it a reasonable baseline for classification tasks.

A closer examination of the Recall metric reveals one of the model's most significant limitations. For the negative class, the model attains an exceptionally high recall of 98%, meaning it successfully identifies nearly all actual negative instances in the dataset, representing a major strength. Conversely, the positive class exhibits a much lower recall of 57%, indicating that the model fails to detect a considerable number of actual positive instances, many of which are either misclassified or overlooked.

Overall, the KNN model exhibits a clear bias toward the negative class, where it performs strongly in both precision and recall. In contrast, it struggles notably with the positive class, particularly in terms of recall, resulting in an unbalanced performance profile. Nevertheless, the model delivers generally good results in overall accuracy, especially for the negative class, where both precision and F1-score remain robust.

	precision	recall	F1-score	support
Negative	78%	98%	87%	184
Positive	96%	57%	71%	116
Accuracy	**82%**			
Marco avg	87%	78%	79%	300
Weighted avg	85%	82%	81%	300

Table 2. KNN Results

Value of k=	Accuracy
3	81%
5	82%
7	81%
11	81%
15	80%

Table 3. KNN with different k

In the case of KNN, we changed the value of (k) to ensure obtaining the best possible value.

4.1.2. Decision tree (DT)

As observed from the table, the Decision Tree model demonstrates a moderately good yet relatively stable performance in the sentiment analysis task. It achieves a

reasonable balance between the negative and positive classes, although noticeable differences exist between the two categories.

Regarding Precision, the model performs similarly across both classes, recording 79% for the negative class and 76% for the positive class. This indicates that most instances predicted as negative or positive are correctly classified, reflecting a relatively balanced precision between classes.

In terms of the F1-score, the model attains 83% for the negative class, showing a strong balance between precision and recall. However, the F1-score decreases to 69% for the positive class, indicating a weaker balance for this category. Despite this decline, the performance remains acceptable, though less stable compared to the negative class.

The model achieves an overall Accuracy of 78%, which is moderate and reflects acceptable general performance. However, this accuracy is lower than that of some other models, suggesting that while the Decision Tree provides reasonable results, it is not the most accurate model available.

A detailed examination of Recall reveals that the model excels in identifying the negative class, achieving 87%, which means it correctly detects most of the actual negative instances. Conversely, the recall for the positive class drops to 64%, indicating that the model fails to identify a substantial portion of positive instances, either misclassifying or overlooking them.

Overall, the Decision Tree model delivers a relatively balanced performance, with a notable tendency to perform better on the negative class than the positive one. While it does not exhibit a strong bias toward a single class, its predictive power remains moderate, particularly in terms of recall for the positive class. Consequently, it can be considered a reasonably good model, albeit limited compared to more advanced approaches.

	Precision	recall	F1-score	support
Negative	79%	87%	83%	184
Positive	76%	64%	69%	116
Accuracy	**78%**			
Marco avg	77%	75%	76%	300
Weighted avg	78%	78%	78%	300

Table 4. decision tree Results

4.1.3. Random forest (RF)

As evidenced by the results presented in the table, the Random Forest model exhibits notably strong and robust performance relative to the previously evaluated models, demonstrating marked improvements across the majority of evaluation metrics. This underscores its efficacy and stability in the context of sentiment analysis tasks.

As evidenced by the results presented in the table, the Random Forest model exhibits notably strong and robust performance relative to the previously evaluated models, demonstrating marked improvements across the majority of evaluation metrics. This underscores its efficacy and stability in the context of sentiment analysis tasks.

Regarding the F1-score, the Random Forest model achieves a commendable balance between Precision and Recall. It records a score of 90% for the negative class, indicating excellent performance in both correctly detecting and classifying negative instances. For the positive class, an F1-score of 80% is observed, reflecting a stable and improved performance relative to preceding models.

In terms of Accuracy, the model attains a high overall score of 87%, surpassing nearly all other models considered in this study. This demonstrates the model's capability to correctly classify the majority of the dataset, positioning it as a reliable and effective tool for sentiment classification.

A more detailed assessment of Recall reveals that the model excels in identifying the negative class, with a recall rate of 98%, indicating that nearly all actual

negative instances are successfully detected. Conversely, the positive class achieves a recall of 68%, which, although improved compared to earlier models, remains suboptimal. This suggests that while some positive instances are missed, the model's performance in this regard is superior to most prior approaches.

In summary, the Random Forest model manifests strong and well-balanced performance, with significant enhancements in overall accuracy and generalization capacity. It consistently outperforms earlier models across multiple evaluation metrics, establishing itself as one of the leading models in this study, particularly in terms of stability and predictive accuracy.

	Precision	recall	F1-score	Support
Negative	78%	98%	90%	184
Positive	96%	68%	80%	116
Accuracy	**87%**			
Marco avg	90%	83%	85%	300
Weighted avg	88%	87%	86%	300

Table 5. Random forest Results

4.1.4. Linear regression (LR)

As indicated by the table, the Linear Regression model (employed here as a classification model) demonstrates robust and well-balanced performance in the sentiment analysis task. It stands out as one of the most stable and consistent models with respect to overall evaluation metrics.

Regarding Precision, the model exhibits high and closely aligned scores across both sentiment classes. It achieves 87% for the negative class, signifying that the majority of negative predictions are accurately classified. For the positive class, it attains a slightly higher precision of 89%, reflecting the model's strong capability to correctly identify positive sentiments with a minimal rate of false positives.

With respect to the F1-score, the model maintains an excellent equilibrium between Precision and Recall. It records a score of 90% for the negative class, indicative of strong and reliable performance in both detecting and classifying negative instances. For the positive class, the F1-score is 83%, which also represents a solid outcome and demonstrates consistent performance when compared to preceding models.

In terms of Accuracy, the model achieves an overall score of 88%, ranking among the highest results across all evaluated models. This indicates the model's proficiency in correctly classifying a substantial proportion of the dataset, reflecting strong generalization and reliability.

A detailed examination of Recall reveals the model's effectiveness in identifying the negative class, with a recall rate of 94%, indicating successful detection of most actual negative instances. The positive class achieves a recall of 78%, which marks a clear improvement over earlier models and signifies enhanced capability in recognizing positive instances without substantial omission.

Overall, the Linear Regression model exhibits highly strong and balanced performance, demonstrating excellent consistency across all evaluation metrics. It does not display significant bias toward either class and maintains stable results fo r both. Consequently, this model can be regarded as one of the most balanced and effective among those assessed in this study.

	Precision	recall	F1-score	Support
Negative	87%	94%	90%	184
Positive	89%	78%	83%	116
Accuracy	**88%**			
Marco avg	88%	86%	87%	300
Weighted avg	88%	88%	87%	300

Table 6. linear regression Results

4.1.5. Adaboost

As observed from the table, the AdaBoost model delivers generally good and consistent performance in the sentiment analysis task, exhibiting relatively balanced results across both classes. However, it does not reach the level of the top-performing models in terms of overall strength or absolute stability.

Regarding Precision, the model demonstrates strong and comparable performance for both classes. It attains 83% for the negative class, indicating that most negative predictions are accurately classified. For the positive class, the precision is slightly higher at 86%, reflecting the model's solid capability to predict positive sentiments with relatively few errors.

In terms of the F1-score, the model shows good performance with an acceptable balance between Precision and Recall. It achieves 88% for the negative class, which indicates strong and stable predictive and detection performance in this category. Conversely, the F1-score for the positive class is 77%, representing a moderate to good outcome and suggesting some disparity in the model's ability to maintain an ideal balance between Precision and Recall compared to the negative class.

Considering Accuracy, the model attains an overall score of 84%, which indicates reasonable competence in correctly classifying the majority of samples. However, this score is lower than that of the best-performing models, positioning AdaBoost as a moderate to good performer in terms of general accuracy.

A deeper analysis of Recall reveals that AdaBoost exhibits strong capability in detecting the negative class, with a recall rate of 93%, meaning that it successfully identifies most of the actual negative instances in the dataset. In contrast, the recall for the positive class is 70%, which is an improvement over some previous models but still indicates that a portion of positive instances are missed or not fully detected.

Overall, the AdaBoost model can be characterized as delivering balanced and good performance overall, with clear superiority in handling the negative class relative to the positive class. It demonstrates acceptable stability across different evaluation metrics, though it remains less powerful than more advanced models in terms of overall accuracy and consistent performance.

	Precision	recall	F1-score	Support
Negative	83%	93%	88%	184
Positive	86%	70%	77%	116
Accuracy	**84%**			
Marco avg	85%	81%	82%	300
Weighted avg	84%	84%	84%	300

Table 7. adaboost Results

4.1.6. XGboost

As observed from the table, the XGBoost model delivers strong and distinguished performance in the sentiment analysis task. It is recognized as an advanced mode l that achieves a good balance between accuracy and stability, exhibiting clear superiority across multiple metrics compared to traditional models.

Regarding Precision, the model demonstrates solid and comparable performance across both classes. It attains 83% for the negative class, indicating that most negative predictions are accurately classified. For the positive class, it achieves a notably high precision of 91%, reflecting the model's strong capacity to predict positive sentiments with high accuracy and a reduced rate of errors.

In terms of the F1-score, the model shows a relatively balanced performance between Precision and Recall. It achieves 89% for the negative class, reflecting strong ability to integrate both precision and recall effectively in this category. Conversely, the positive class records an F1-score of 78%, which, while good, is somewhat lower than that of the negative class, suggesting some disparity between the two classes despite remaining within a robust range.

With respect to Accuracy, the model attains an overall score of 85%, indicating a strong ability to correctly classify the majority of samples, thereby positioning it as an effective and reliable model for sentiment analysis tasks.

A more detailed examination of Recall reveals that the model performs excellently in the negative class, achieving a recall rate of 96%, which signifies near-complete detection of actual negative instances in the dataset. In contrast, the positive class recall is 68%, a respectable figure that indicates the model misses some positive instances, although it still outperforms several traditional models in this aspect.

Overall, the XGBoost model can be characterized as delivering strong and competitive performance, with good stability and high generalization capability, particularly for the negative class. It is among the advanced models that yield robust results for sentiment analysis, despite some imbalance in recall performance between the classes.

	Precision	recall	F1-score	support
Negative	83%	96%	89%	184
Positive	91%	68%	78%	116
Accuracy	**85%**			
Marco avg	87%	82%	83%	300
Weighted avg	86%	85%	84%	300

Table 8. XGboost Results

4.2. Deep learning

We applied the same evaluation process to the previously mentioned deep learning models, and their results are summarized in the tables below.

4.2.1. Convolutional Neural Network (CNN) :

As observed from the table, the Convolutional Neural Network (CNN) model exhibits an unbalanced performance in the sentiment analysis task, demonstrating clear strength in detecting the negative class but significant weakness in handling the positive class. This imbalance results in a lower overall performance compared to other models.

In terms of Precision, the model achieves very strong performance for the positive class, recording 96%, which indicates that the majority of positive predictions are accurate and reliable. Conversely, for the negative class, precision is lower at 73%, suggesting a higher rate of errors when predicting negative instances compared to the positive class.

Regarding the F1-score, the model shows a marked disparity between the two classes. It attains 84% for the negative class, reflecting a reasonable ability to detect and classify negative instances. However, the F1-score drops substantially to 57% for the positive class, indicating a pronounced imbalance between precision and recall and an unstable performance in identifying positive sentiments.

With respect to Accuracy, the model achieves an overall score of 76%, which is moderate to low relative to other models. This reflects the CNN model's limited capacity to deliver strong general performance in this particular task.

A more detailed analysis of Recall reveals that the model excels in the negative class, with a recall rate of 99%, signifying near-complete detection of actual negative instances and demonstrating high sensitivity to this category. In contrast, the positive class recall is only 41%, a very low figure that indicates the model fails to detect a significant portion of positive instances, leading to substantial loss or misclassification of positive data.

Overall, the CNN model suffers from a pronounced imbalance in performance, heavily biased toward the negative class at the expense of the positive class. Despite its strengths in certain areas, the overall performance remains constrained compared to both traditional and advanced models in this study, particularly due to its poor recall for the positive class.

	Precision	recall	F1-score	support
Negative	73%	99%	84%	184
Positive	96%	41%	57%	116
Accuracy	**76%**			

Marco avg	84%	70%	70%	300
Weighted avg	82%	76%	73%	300

Table 9. CNN Results

4.2.2. RNN-Long Short-Term Memory (LSTM) :

As observed from the table, the Long Short-Term Memory (LSTM) model demonstrates strong and distinguished performance in the sentiment analysis task. It is among the most stable and balanced models evaluated, particularly effective when handling context-dependent textual data.

In terms of Precision, the model achieves very high performance for the negative class, recording 93%, which indicates that the majority of negative predictions are highly accurate. For the positive class, it attains a respectable precision of 82%, reflecting an acceptable to strong ability to predict positive sentiments with relatively few errors.

Regarding the F1-score, the model displays a robust balance between Precision and Recall. It achieves an excellent score of 91% for the negative class, demonstrating a high capability to integrate precision and recall effectively. For the positive class, the F1-score is 84%, representing very good performance and clear stability in predicting this class compared to many other models.

With respect to Accuracy, the model attains a high overall score of 89%, ranking among the best results across all tested models. This reflects a strong ability to correctly classify the majority of samples and indicates reliable and balanced general performance.

A detailed analysis of Recall reveals that the model performs very well in the negative class, with a recall rate of 90%, indicating effective and accurate detection of most negative instances. For the positive class, recall reaches 87%, a strong figure that highlights the model's excellent capability to identify the majority of positive cases without significant loss, marking a clear improvement over many previous models.

Overall, the LSTM model delivers highly strong and balanced performance, exhibiting a superior capacity to understand textual context and handle both classes

comparably. It demonstrates clear stability across all evaluation metrics, making it one of the top-performing models in this study in terms of accuracy and balance, though there remains potential for further enhancement.

	Precision	recall	F1-score	support
Negative	93%	90%	91%	196
Positive	82%	87%	84%	104
Accuracy	**89%**			
Marco avg	87%	88%	88%	300
Weighted avg	89%	89%	89%	300

Table 10. LSTM Results

4.2.3. RNN-Gated Recurrent Units (GRU) :

As observed from the table, the Gated Recurrent Unit (GRU) model delivers strong and consistent performance in the sentiment analysis task. It is among the deep learning models that achieve a good balance between precision and recall, demonstrating clear stability across both classes.

In terms of Precision, the model performs at a high and comparable level for both classes. It achieves 87% for the negative class, indicating that most negative predictions are accurately classified. For the positive class, precision is slightly higher at 90%, reflecting the model's strong capability to predict positive sentiments with high accuracy and minimal classification errors.

Regarding the F1-score, the model exhibits very strong and balanced performance between precision and recall. It attains an excellent score of 91% for the negative class, reflecting a high ability to integrate precision and recall effectively. For the positive class, the F1-score is 84%, which is also strong and stable compared to many other models, indicating an acceptable balance in handling both classes.

With respect to Accuracy, the model achieves an overall score of 88%, reflecting a strong ability to correctly classify most samples. This positions the GRU among the advanced models in terms of overall performance and stability.

A detailed analysis of Recall reveals that the model demonstrates excellent capability in detecting the negative class, with a recall rate of 95%, indicating accurate recognition of most negative instances within the dataset. The positive class recall is 87%, a very good figure that indicates the model's strong ability to effectively detect positive instances with only a small number of missed cases.

Overall, the GRU model delivers highly strong and balanced performance, with clear stability across all evaluation metrics and a good capacity for understanding textual context. It is considered one of the best-performing DL models in this study, owing to its well-balanced precision and recall across both classes, although it is not yet the absolute best.

	precision	recall	F1-score	support
Negative	87%	95%	91%	184
Positive	90%	87%	84%	116
Accuracy	**88%**			
Marco avg	89%	87%	87%	300
Weighted avg	88%	88%	88%	300

Table 11. GRU Results

	Precision	recall	F1-score	Support
Negative	94%	91%	92%	196
Positive	84%	88%	86%	104
Accuracy	**90%**			

Marco avg	89%	90%	89%	300
Weighted avg	90%	90%	90%	300

Table 12. SN.MODEL Results

4.3. SN.MODEL

4.3.1. DZIRIBERT:

At the beginning of the process, the DziriBERT model a Transformer-based architecture tailored specifically for Arabic and the Algerian dialect is employed. Unlike models that directly perform classification, DziriBERT's primary role is to convert raw text into rich numerical representations known as embeddings.

For instance, when given a sentence like "هذا فيلم شباب" DziriBERT does not simply treat it as a sequence of independent words. Instead, it analyzes the relationships between words within the full context of the sentence. The input text is first tokenized into smaller units, which are then passed through multiple Transformer layers. These layers use the Self-Attention mechanism to assess how each word relates to every other word in the sentence, capturing nuanced contextual information.

The output of this process is a matrix often with dimensions such as (32, 768) where each token corresponds to a vector of 768 features that encapsulate its contextual meaning. In this way, DziriBERT transforms unstructured textual data into a rich, structured mathematical form that downstream models can effectively use. Thus, DziriBERT functions as a powerful feature extractor, providing deep language understanding rather than performing direct classification.[fig 14.]

```
Algorithm: Get Embedding
Input:
   text
Output:
   embedding vector
Begin
   Convert text to tokens
   Apply padding and truncation
   Pass tokens to BERT model
   Extract hidden state
   Return embedding
```

Fig 14. Dziribert pseudo code

4.3.2. BILSTM (Bidirectional Long Short-Term Memory) :

After obtaining the embeddings from DziriBERT, they are fed into a BiLSTM (Bidirectional Long Short-Term Memory) model. This type of Recurrent Neural Network (RNN) is designed to process sequences in both directions forward (from beginning to end) and backward (from end to beginning) allowing it to capture richer temporal dependencies within the text [31].

In this system, the BiLSTM's role is to deepen the understanding of the sequence of embeddings by modeling how the meaning evolves depending on the order of words, complementing the contextual information captured by DziriBERT. It dynamically processes the token embeddings to build a nuanced representation of the sentence. [fig 15.]

Following the BiLSTM, an Attention Pooling layer is applied. This mechanism highlights the most important words in the sentence by assigning them higher weights during feature aggregation. For example, emotionally charged words like "مليح" or "سيء" receive greater emphasis compared to less informative words, enhancing the model's focus on key sentiment signals.

Ultimately, the BiLSTM combined with attention produces a feature vector a compact and informative representation of the sentence that encapsulates its essential semantic content. This feature vector is then used for the final classification task.

```
Algorithm: Supervised BiLSTM Model
Input:
   input features
Output:
   predicted class
Begin
   Pass input to BiLSTM
   Apply Attention Pooling
   Extract features
   Pass features to classifier
   Return prediction
End
```

Fig 15. BiLSTM pseudo Code

4.3.3. SMOTE :

After extracting features from the BiLSTM model, a common challenge in sentiment analysis emerges: class imbalance. This occurs when one sentiment class, such as positive sentences, vastly outnumbers another, like negative sentences, causing the model to become biased toward the majority class.

To address this, the SMOTE (Synthetic Minority Over-sampling Technique) algorithm is employed. Unlike simple duplication of minority class samples, SMOTE generates new synthetic examples to enrich the minority class.

SMOTE operates by identifying nearby data points within the feature space and creating new samples along the lines connecting these neighbors through linear interpolation. Essentially, if two data points represent similar minority instances, SMOTE generates additional points between them, effectively expanding the minority class region. [fig 16.]

This approach balances the class distribution, enabling the model to learn more equally from both classes, which enhances overall performance and reduces bias in predictions [32].

```
SMOTE with k=3, seed=42
X_bal, y_bal ← smote.fit(X_lstm, y_train)
 scale_pos_weight ← count(y=0) / count(y=1)
```

Fig 16. SMOTE pseudo Code

4.3.4. XGBOOST CLASSIFIER :

After obtaining the balanced and processed data, it is passed to the XGBoost model, which is widely regarded as one of the most powerful ML algorithms based on Gradient Boosting.

XGBoost builds a sequence of decision trees iteratively, where each new tree focuses on correcting the errors made by the previous ones. This approach makes the model highly effective at capturing complex, non-linear patterns in data.

Instead of using raw text, the features extracted by the BiLSTM model serve as input to XGBoost. The primary role of XGBoost here is to perform the final classification, categorizing the input into positive or negative sentiment. To enhance performance and prevent overfitting, techniques such as regularization and subsampling are applied, improving the model's ability to generalize to unseen data.

By integrating BiLSTM's deep feature extraction with XGBoost's robust classification capabilities, the system effectively balances rich representation learning with strong predictive accuracy. [fig 17.]

```
XGB ← XGBClassifier(
trees=1000, depth=5, lr=0.05,
subsample=0.8, col_sample=0.8,
 α=0.1, λ=1.5, w=scale_pos_weight,
 metric=logloss, early_stop=30, seed=42
 )
XGB.fit(X_bal, y_bal, eval=(X_test, y_test))
```

Fig 17. XGboost pseudo Code

4.3.5. FRONTEND :

The final stage of the system is the user interface, developed using Flask [33]. This interface serves as a bridge between the model and the end user, providing an accessible and visually appealing platform for interaction. The background features an image of Algeria's capital city, with subtle adjustments to color and brightness to enhance aesthetics without distracting from the content.

At the top, a text input box allows users to enter sentences, accompanied by a display of the languages supported by the system. Below the input, a button initiates the sentiment analysis process. [fig 18.]

When a user submits a sentence, Flask sends the text to the underlying model for analysis. The model processes the input to determine whether the sentiment is positive [fig 19.] or negative [fig 20.]. The result is then returned and displayed on the

interface, showing the predicted sentiment alongside a circular percentage.

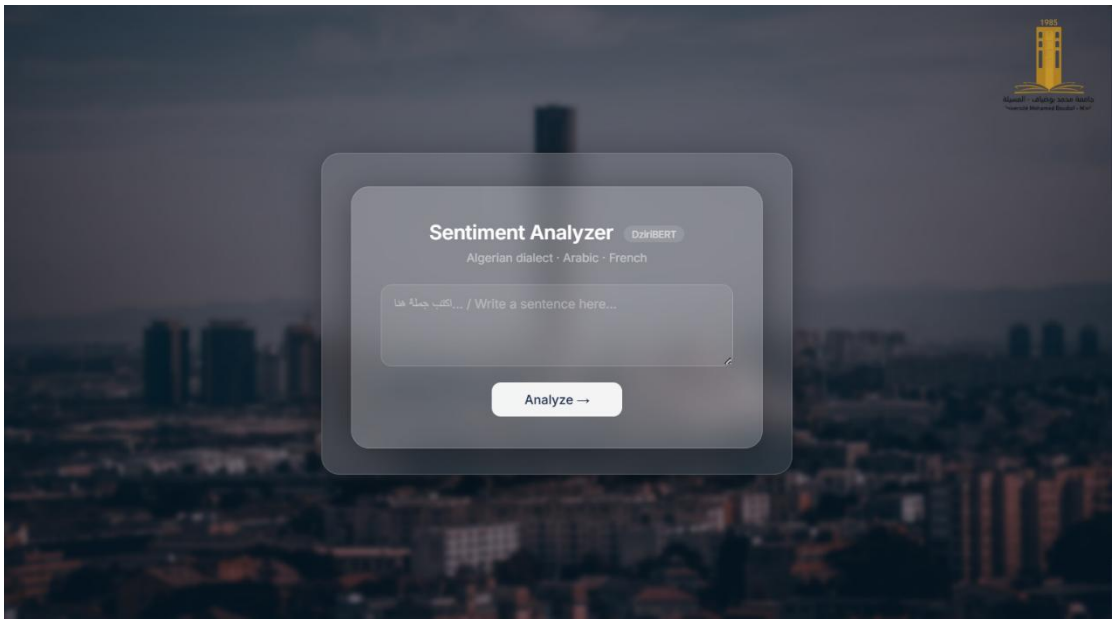

Fig 18. Frontend in web

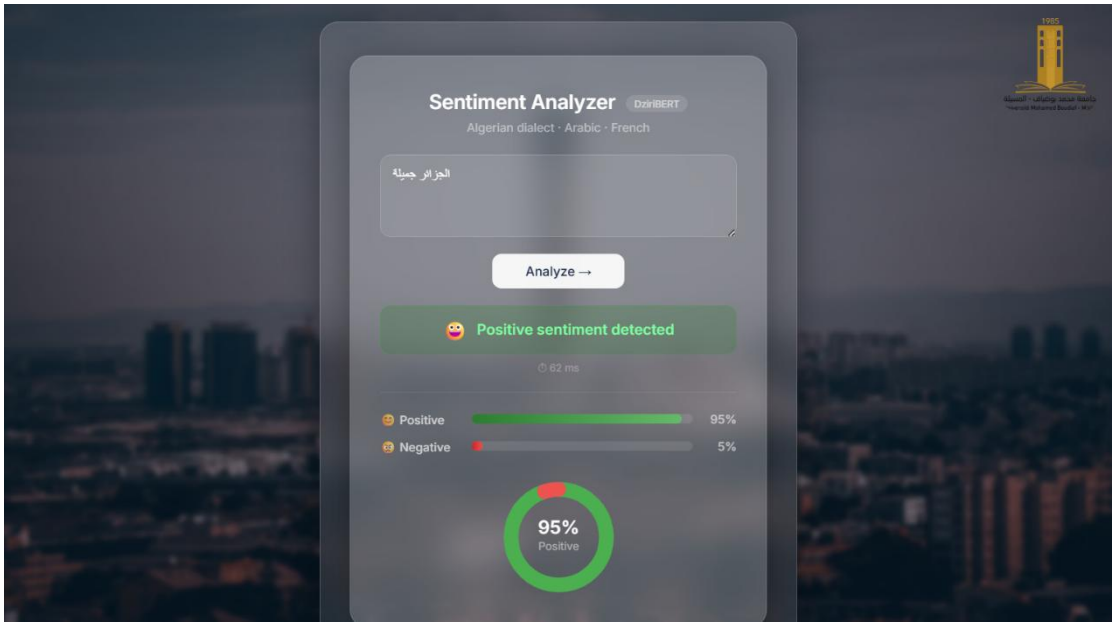

Fig 19. The Frontend in positive

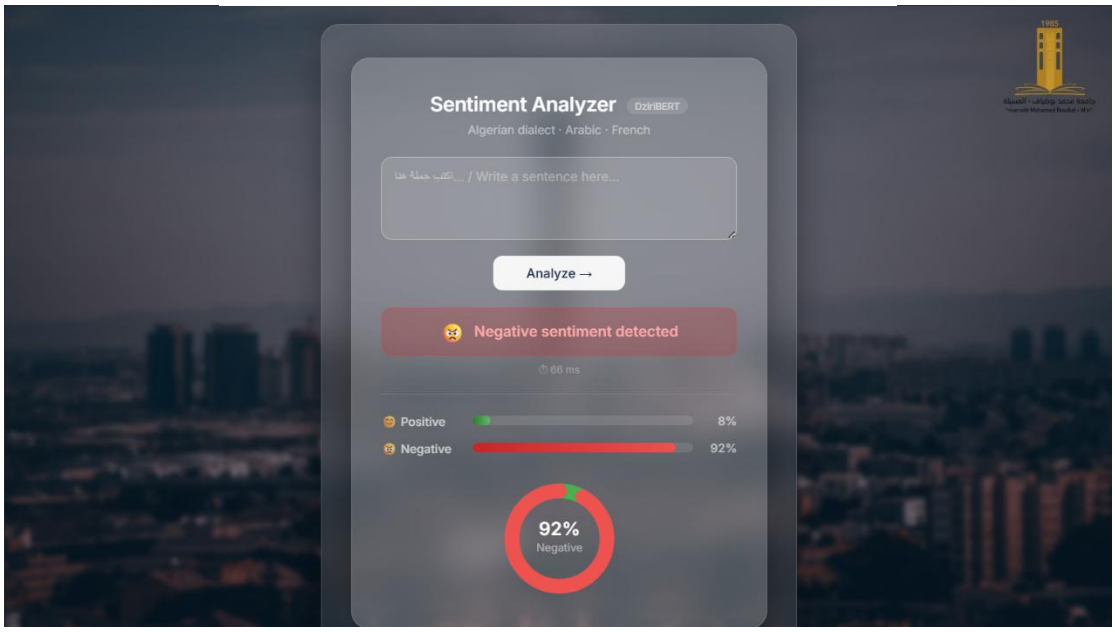

Fig 20. The Frontend in negative

Conclusion

Artificial Intelligence plays a significant role in our daily lives, prompting many countries to accelerate research and development in this field due to its expanding relevance across various domains. As outlined in this dissertation, we aimed to contribute to this area by focusing on sentiment analysis of the Algerian dialect within social media texts. This effort led to the development of a hybrid model combining BiLSTM, DziriBERT, and XGBoost, designed to effectively process and analyze sentiments expressed in Algerian dialect texts.

Our proposed model demonstrated notable improvements over several other approaches such as K-Nearest Neighbors (KNN), Random Forest, and Convolutional Neural Networks (CNN). It achieved strong performance metrics including Precision, Recall, F1-score, and Accuracy reaching approximately 90% accuracy. These results underscore the model's effectiveness despite the complexities of the Algerian dialect, its linguistic diversity, and the varied ways users express it online.

The study also emphasizes the importance of modern Natural Language Processing techniques, particularly for Arabic dialects that still require extensive research and resources. Looking ahead, we intend to further enhance the model, expand the dataset, and improve results to offer more accurate and objective sentiment analysis for users.

Our future objectives for improving the program include the following:

- Enhancing and strengthening the model by incorporating additional data to improve performance and robustness.
- Expanding the program's capabilities to analyze multimodal data, including images and audio, and generate corresponding sentiment analysis results.
- Implementing techniques to identify specific words or sentences that contribute to the prediction outcomes, thereby improving model interpretability.
- Introducing real-time sentiment analysis functionality, particularly for dynamic data streams from social media platforms.
- Optimizing data processing speed and reducing response time to achieve faster and more efficient results.
- Developing the system for deployment as a mobile application and as a fully integrated web application to increase accessibility and usability.

References

[1] Carthy, J., Minsky, M., Rochester, N., Shannon, C.E., "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence"., http://raysolomonoff.com/dartmouth/boxa/dart564props.pdf August, 1955

[2] McCarthy, John; Minsky, Marvin; Rochester, Nathan; Shannon, Claude (1955), A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, archived from the original on 2007-08-26, retrieved 2006-04-09 retrieved 10:47 (UTC), 9th of April 2006

[3] S. Solomonoff, R.J. "The Time Scale of Artificial Intelligence; Reflections on Social Effects", Human Systems Management, Vol 5, pp. 149–153, 1985

[4] Adam Westerski "Sentiment Analysis: Introduction and the State of the Art overview",Universidad Politecnica de Madrid, Spain,2003

[5] Omar F. Zaidan , Chris Callison-Burch "Arabic Dialect Identification", Association for Computational Linguistics , vol 40 , No 1 , pp 172-202 , 2014

[6] مبارك محمد الميلي ."تاريخ الجزائر القديم و الحديث" .

[7] Sonit Singh "Natural Language Processing for Information Extraction",Department of Computing, Faculty of Science and Engineering, Macquarie University, Australia

[8] Zhongheng Zhang “Introduction to machine learning: k-nearest neighbors” nnals of Translational Medicine , Ann Transl Med , Vol 4 , No 11 2016 .

[9] Soudamini Hota , Sudhir Pathak “KNN classifier based approach for multi-class sentiment analysis of twitter data"International Journal of Engineering & Technology, vol 7 , N 3 , 2018 , pp.1372-1375

[10] H KIM , GJ KOEHLER “Theory and Practice of Decision Tree Induction" Omega, Int. J. Mgmt Sci. Vol. 23, No. 6, pp. 637-652, 1995

[11] Achmad Bayhaqy ,and al “Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes" International Conference on Orange Technologies , Conference : 2018 Bali, Indonesia , DOI: 10.1109/ICOT.2018.8705796 .

[12] Betina P.O. Lovatti , and al “Use of Random forest in the identification of important variables" Microchemical Journal , Vol 145 , pp 1129-1134 , 2019

[13] Yassine Al Amrani , Mohamed Lazaar , Kamal Eddine El Kadiri "Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis" Procedia Computer Science vol 127 (2018) , pp. 511–520 [Online].Available :www.sciencedirect.com

[14] Astrid Schneider , and al "Linear Regression Analysis" Procedia - Social and Behavioral Sciences , Vol 106 , pp 234 – 240 , 2013

[15] Nikhil Kumar Singh · Deepak Singh Tomar · Arun Kumar Sangaiah , "Sentiment analysis: a review and comparative analysis over social media" Journal of Ambient Intelligence and Humanized Computing 2020 N 11 , pp. 97–117 ,[Online] available : https://doi.org/10.1007/s12652-018-0862-8

[16] M. Kawakita , and al "An introduction to the predictive technique AdaBoost with a comparison to generalized additive models" Preprint submitted to Elsevier Science , 18 july 2005

[17] Vladimir Markovic ,and al" Employee reviews sentiment classification using BERT encoding and AdaBoost classifier tuned by modified PSO algorithm" Advances in Computer Science Research , pp. 113,[Online] available : https://doi.org/10.2991/978-94-6463-482-2_3

[18] Ramraj S , and al "Experimenting XGBoost Algorithm for Prediction and Classifi cation of Different Datasets" International Journal of Control Theory and Applications Vol 9, No 40, 2016

[19] I Gusti Ayu Nandia Lestari , and al"Effectiveness of AdaBoost and XGBoost Algorithms in Sentiment Analysis of Movie Reviews" This paper has been accepted by IEEE Transactions on Cybernetics, DOI:10.1109/TCYB.2020.2983860

[20] Yanan Sun , and al"Automatically Designing CNN Architectures Using Genetic Algorithm for Image Classification " IEEEAccess vol 9, 2021

[21] Qiannan Zhu , Xiaofan Jiang , Renzhen Ye , "Sentiment Analysis of Review Text Based on BiGRU-Attention and Hybrid CNN" IEEEAccess vol 9, 2021

[22] Ralf C. Staudemeyer , Eric Rothstein Morris "Understanding LSTM a tutorial into Long Short-Term Memory Recurrent Neural Networks" [online] : arXiv:1909.09586v1 [cs.NE] 12 Sep 2019

[23] Dr. G. S. N. Murthy ,and al "Text based Sentiment Analysis using LSTM", International Journal of Engineering Research & Technology (IJERT) Vol. 9 , 2020

[24] Saeed Mohsen "Recognition of human activity using GRU deep learning algorithm" Multimedia Tools and Applications , Vol 82 , pp 47733–47749 , 2023 , [online] : https://doi.org/10.1007/s11042-023-15571-y

[25] WAQAR ALI , and al "Aspect-Level Sentiment Analysis Based on Bidirectional-GRU in SIoT" IEEEAccess , vol 9 , 2021

[26] Sumeshwar Singh "BERT Algorithm used in Google Search" Mathematical Statistician and Engineering Applications , Vol 70 , No 2 , pp 1641-1650, 2021.

[27] Mickel Hoang , and al"Aspect-Based Sentiment Analysis Using BERT" Proceedings of the 22nd Nordic Conference on Computational Linguistics , pp 187–196, September–October 2019

[28] Amira Hamed , and al , "Embedding Extraction for Arabic Text Using the AraBERT Model" Computers, Materials & Continua , Tech Science Press , 17 January 2022 , DOI: 10.32604/cmc.2022.025353

[29] Wissam Antoun , Fady Baly , Hazem Hajj , "AraBERT: Transformer-based Model for Arabic Language Understanding" Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pp. 9-15

[30] Amine Abdaou ,and al "DziriBERT: a Pre-trained Language Model for the Algerian Dialect" [online] available : https://www.researchgate.net/publication/354894266_DziriBERT_a_Pre-trained_Language_Model_for_the_Algerian_Dialect

[31] Zabit hameed , Begonya- zapirain "Sentiment Classification Using a Single-Layered BiLSTM Model" IEEEAccess , Vol 8 , pp 73992-74001, April 17, 2020

[32] Rok Blagus and Lara Lusa "SMOTE for high-dimensional class-imbalanced data", Blagus and Lusa BMC Bioinformatics . pp 1-16 , 2013 .

[33] Devndra Ghimire“Comparative study on Python web frameworks: Flask and Django”, Metropolia University of Applied Sciences Bachelor of Engineering Media Engineering , Bachelor’s Thesis , 5 May 2020