Democratic and Popular Republic of Algeria
الجمهورية الجزَائِرية الديمقرَاطية الشعبية
Ministry of Higher Education and Scientific Research
وزَارة التعليم العَالي و البحث العلمي

University Center El Cherif Bouchoucha Aflou
المرَكَز الجَامِعي الشَريف بُوشُوشَة آفلُو
Science Institute
مَعهَد العُلُوم
Departement of Computer Science
قِسم الاَعلَام الآلي

Presented to obtain the degree of
Licence LMD in Computer Science

EmoMetrix: A Machine Learning-enabled Text-based Sentiment Analyzer for the Aflou Region Dialect

Presented by :
Ms. ZERROUK Hadil Ines
Ms. BENDEBLA Imane

Supervised by:
Mr. BENMOUSSA Ahmed

In front of the jury composed of :

Mr. AZZOUZI Oussama	MAA	UC-Aflou	President
Mr. LAIB Lakhdar	MCB	UC-Aflou	Examiner
Mr. BENMOUSSA Ahmed	MCB	UC-Aflou	Advisor

2022/2023

Acknowledgements

We would like to express our heartfelt gratitude to ALLAH for granting us the strength and health to complete this work.

We extend our sincere appreciation to our supervisor, Dr Mr. Ben Moussa Ahmed, for his unwavering support, guidance, and availability throughout our project. His expertise and willingness to assist us have been invaluable in our progress.

We are also immensely grateful to Dr Mr. Laib Lakhdar, Dr Mr. Sellam Abdellah, and Dr Mr. Ben Hessine Nessredine for their exceptional efforts in organizing and providing the workshop of artificial intelligence. Their dedication and expertise have greatly contributed to our knowledge and skills.

A special thank you goes to Dr Mrs. Charef Fatiha for her invaluable assistance in providing the dataset for our project. Her contribution has been instrumental in the development and success of our work.

We would like to express our gratitude to Dr Mrs. Bousbouaa Fatima for his support and training in utilizing Overleaf, which has enhanced the presentation of our report.

Furthermore, we are immensely grateful to Prof. Dr. Tahari Abdelkarim, our university director and professor, as well as the head of the Institute Dr Mr Ben Sassi Ahmed and the department chief, Dr. Mr. Azzouzi Ousama, for their efforts in organizing the week of artificial intelligence. Their support has been crucial in facilitating the completion of our project.

Lastly, we would like to extend our sincere thanks to all our professors at the university for their knowledge, mentorship, and encouragement, which have played a vital role in shaping our academic journey and fostering our personal growth.

Dedication

To my cherished parents, the source of life, love, and affection,
To my brave brothers Ahmed and Amjed my true heroes,
To my beloved sisters Lina, Maram, Ranim, and Bissan the guiding lights that illuminate my path,
To my dear friends Aymen and Bouchra who have been by my side through thick and thin,
To the members of the 10Brains Club, my constant motivation to work and progress always,
And to all my esteemed professors who have played a significant role in shaping my university career
-Arigato-

Hadil

I dedicate this project to Allah Almighty my creator, my strong pillar,
my source of inspiration, wisdom, knowledge and understanding
With my deepest feelings of gratitude, I dedicate My dearest
mother & To my dearest father My deepest thanks To all my family members
To all my friends for the wonderful times I had with them at
university and To all the teachers who were the reason for our success
to those who love us and those whom we love, to those who have left us and
have left their good impact on our hearts, may Allah have mercy on them

Imane

المَلخص

تحليل المشاعر هو أداة قوية يمكنها توفير رؤى قيمة حول مواقف وآراء الأفراد والمجموعات والمجتمعات. من خلال تحليل المشاعر تلقائيًا لبيانات النصوص، يمكن لتحليل المشاعر مساعدة الشركات والمؤسسات والأفراد في فهم الطابع العاطفي للمنشورات على وسائل التواصل الاجتماعي وآراء العملاء وغيرها من أشكال الاتصال المستندة إلى النصوص. على الرغم من أن تحليل المشاعر قد درس على نطاق واسع للغات الرئيسية مثل الإنجليزية والصينية والإسبانية، إلا أن هناك العديد من اللغات واللهجات التي تفتقر إلى مكتبات محللي المشاعر الجاهزة. في هذا السياق، فإن تطوير خوارزميات وتقنيات جديدة لتحليل المشاعر أمر بالغ الأهمية لتقدم مجال معالجة اللغة الطبيعية ودعم اللغات واللهجات التي تعاني من نقص الموارد.

في هذا المشروع، نهدف إلى تطوير خوارزمية لتحليل المشاعر تتماشى مع لهجتنا، التي تفتقر حاليًا إلى أي مكتبات محللة للمشاعر مسبقة الوجود. لهجتنا لها خصائص لغوية وثقافية فريدة تجعل من الصعب تطوير أدوات فعالة لتحليل المشاعر، وبالتالي يمثل مشروعنا مساهمة كبيرة في مجال معالجة اللغة الطبيعية. باستخدام تقنيات التعلم الآلي ومعالجة اللغة الطبيعية، سنقوم بتدريب خوارزميتنا على مجموعة ضخمة من البيانات الموسومة، مما سيتيح لها تصنيف النصوص بدقة بالمشاعر في لهجتنا. من خلال هذا المشروع، نأمل في التصدي للحاجة الملحة لأدوات تحليل المشاعر في لهجتنا والمساهمة في الهدف الأوسع لدعم اللغات واللهجات التي تعاني من نقص الموارد في مجال معالجة اللغة الطبيعية.

الكلمات المفتاحية: تحليل المشاعر، معالجة اللغة الطبيعية، التعلم الآلي، اللغة العربية

Abstract

Sentiment analysis is a powerful tool that can provide valuable insights into the attitudes and opinions of individuals, groups, and communities. By automatically analyzing the sentiment of text data, sentiment analysis can help businesses, organizations, and individuals understand the emotional tone of social media posts, customer reviews, and other forms of text-based communication. While sentiment analysis has been widely studied for major languages such as English, Chinese, and Spanish, there are many languages and dialects that lack pre-existing sentiment analysis libraries. In this context, developing new algorithms and techniques for sentiment analysis is crucial for advancing the field of natural language processing and supporting under-resourced languages and dialects.

In this project, we aim to develop an algorithm for sentiment analysis that is tailored to our dialect, which currently lacks any pre-existing sentiment analysis libraries. Our dialect has unique linguistic and cultural characteristics that make it challenging to develop effective sentiment analysis tools, and so our project represents a significant contribution to the field of natural language processing. By using machine learning and natural language processing techniques, we will train our algorithm on a large corpus of labeled data, allowing it to accurately classify the sentiment of text data in our dialect. Through this project, we hope to address the pressing need for sentiment analysis tools in our dialect, and contribute to the broader goal of supporting under-resourced languages and dialects in the field of natural language processing.

Keywords: sentiment analysis, natural language processing, Machine learning, Arabic language.

Résumé

L'analyse des sentiments est un outil puissant qui peut fournir des informations précieuses sur les attitudes et opinions des individus, des groupes et des communautés. En analysant automatiquement le sentiment des données textuelles, l'analyse des sentiments peut aider les entreprises, les organisations et les individus à comprendre le ton émotionnel des publications sur les médias sociaux, des avis clients et d'autres formes de communication basées sur du texte. Alors que l'analyse des sentiments a été largement étudiée pour les grandes langues telles que l'anglais, le chinois et l'espagnol, de nombreuses langues et dialectes ne disposent pas de bibliothèques d'analyse des sentiments préexistantes. Dans ce contexte, le développement de nouveaux algorithmes et techniques pour l'analyse des sentiments est crucial pour faire avancer le domaine du traitement automatique du langage naturel et soutenir les langues et dialectes sous-ressourcés.

Dans ce projet, notre objectif est de développer un algorithme d'analyse des sentiments adapté à notre dialecte, qui ne dispose actuellement d'aucune bibliothèque d'analyse des sentiments préexistante. Notre dialecte présente des caractéristiques linguistiques et culturelles uniques qui rendent difficile le développement d'outils efficaces d'analyse des sentiments, et notre projet représente donc une contribution significative au domaine du traitement automatique du langage naturel. En utilisant des techniques d'apprentissage automatique et de traitement automatique du langage naturel, nous allons entraîner notre algorithme sur un grand corpus de données étiquetées, lui permettant de classifier avec précision le sentiment des données textuelles dans notre dialecte. À travers ce projet, nous espérons répondre au besoin urgent d'outils d'analyse des sentiments dans notre dialecte et contribuer à l'objectif plus large de soutenir les langues et dialectes sous-ressourcés dans le domaine du traitement automatique du langage naturel.

Mots clés: analyse des sentiments, traitement automatique du langage naturel, apprentissage automatique, langue arabe.

Abbreviations list

NLP: Natural Language Processing
GPT: Generative Pretrained Transformer
BERT: Bidirectional Encoder Representations from Transformers
AI: Artificial Intelligence
ML: Machine learning
DL: Deep learning
MaxEnt: Maximum Entropy
SVM: Support Vector Machines
CNN: Convolutional Neural Network
RNN: Recurrent Neural Networks
DBN: Deep Belief Networks
LSTM: Long Short-Term Memory
GRU: Gated Recurrent Unit
GPU: Graphics Processing Units
TPU: Tensor Processing Units
NB: Naive Bayes
POS: Part of Speech
VS Code: Visual Studio Code
HTML: Hypertext Markup Langage
CSS: Cascading Style Sheets
SQL: Structured Query Langage
PNG: Protocol Network Graphics

CONTENTS

LIST OF FIGURES

CHAPTER

1

GENERAL INTRODUCTION

In recent years, the exponential increase in the use of the internet and the exchange of opinions and sentiments expressed in various ways, including the amount of detail given, the type of vocabulary used, the writing context, slang, and linguistic variations, has made manual sentiment analysis tedious and almost impossible. Therefore, sentiment analysis has become a crucial area of research in natural language processing. Sentiment classification is the essential step of sentiment analysis, which is a traditional text classification problem that classifies documents of different subjects such as science, sports and politics,. However, in sentiment classification, words expressing sentiment or opinion that indicate positive or negative opinions are more important. Opinion mining is a task that extracts opinions from a set of relevant documents for a given subject. Sentiment (or opinion) can be expressed in various subtle ways, making it difficult to determine. Sentiment classification (polarity) is a subtask of opinion mining and generally involves determining whether the document's opinion on the subject is positive or negative. Opinion mining can be performed at the document, paragraph, or sentence level.

CHAPTER

2

STATE OF THE ART

2.1 Introduction

The field of sentiment analysis is rapidly growing and focuses on understanding the emotions, opinions, and attitudes behind a piece of text, whether it is positive, negative, or neutral. In today's digital age, where individuals express their thoughts and feelings through social media, blogs, and product reviews, businesses need to understand customer feedback and improve their products and services accordingly.

This chapter provides an overview of sentiment analysis, including its significance, processes, and approaches. By the end of the chapter, readers will have a strong understanding of the fundamentals of sentiment analysis and its importance in today's world.

2.2 Overview about "NLP"

Natural Language Processing (NLP) is a field of study that focuses on the interaction between computers and human language. It involves using computational techniques to analyze, understand, and generate human language. NLP is a multidisciplinary field that draws on knowledge from computer science, linguistics, mathematics, psychology, and other related fields. NLP has many applications, including but not limited to:

1. Sentiment Analysis: determining attitude/emotion towards a topic.
2. Machine Translation: translating text automatically between languages.
3. Information Extraction: extracting structured information from unstructured text.
4. Question Answering: answering questions automatically in natural language.
5. Text Summarization: creating shorter version of text while retaining important info.

NLP has made progress with machine learning, deep learning, and AI. GPT-3 and BERT are notable breakthroughs. It has potential to transform many industries, and is increasingly important for extracting insights from text data.

2.3 Definition of sentiment analysis

Sentiment analysis, also known as opinion mining, is a subfield of natural language processing that aims to identify and extract subjective information from text data, such as opinions, emotions, and attitudes. According to [5] Sentiment analysis refers to the use of natural language processing, text analysis, computational linguistics, and other statistical and machine learning techniques to identify, extract, and quantify the emotional states, attitudes, opinions, and subjective information expressed in text, speech, or other forms of unstructured data. Also [13] define sentiment analysis as a computational technique used to determine the sentiment expressed in a piece of text, and it is based on the identification and extraction of relevant features from the text, such as words, phrases, or emotions, which are then mapped to a specific sentiment or emotion.

2.4 Importance of sentiment analysis

Sentiment analysis has become an essential tool in various fields, including business, politics, and social media. According to the author's of [2] opinion Sentiment analysis is necessary for various reasons, particularly in today's digital world where individuals, organizations, and governments generate massive amounts of text data through social media platforms, emails, surveys, and online reviews. Some of the main reasons why sentiment analysis is necessary are:

1. Business Intelligence: it helps businesses understand customer opinions and feedback, improve products and services, and develop effective marketing strategies.

2. Reputation Management: it helps organizations monitor and manage their online reputation by analyzing the sentiments of customers, competitors, and stakeholders.

3. Customer Service: it helps organizations identify customer issues and respond proactively to improve satisfaction and loyalty.

4. Political Campaigns: sentiment analysis helps political parties understand public opinions and attitudes towards parties and candidates, and develop effective strategies to engage with voters.

5. Social Media Analysis: it is necessary to monitor and analyze individuals' sentiments towards events, celebrities, brands, and products.

2.5 The process of sentiment analysis

1. Data Collection: Gathering data from various sources such as social media, customer feedback forms, and product reviews in various formats, including text, audio, or video.

2. Data Preprocessing: Cleaning, transforming, and standardizing collected data by removing stop words, punctuation marks, and special characters for analysis.

3. Feature Extraction: Identifying and extracting relevant features from preprocessed data using techniques such as tokenization, part-of-speech tagging, and named entity recognition for sentiment analysis.

4. Sentiment Classification: Categorizing text sentiment as positive, negative, or neutral using extracted features with techniques and approaches .

5. Sentiment Analysis Visualization: Visualizing sentiment analysis results in a graphical format for easy interpretation using data visualization tools like word clouds, bar charts, and scatter plots.

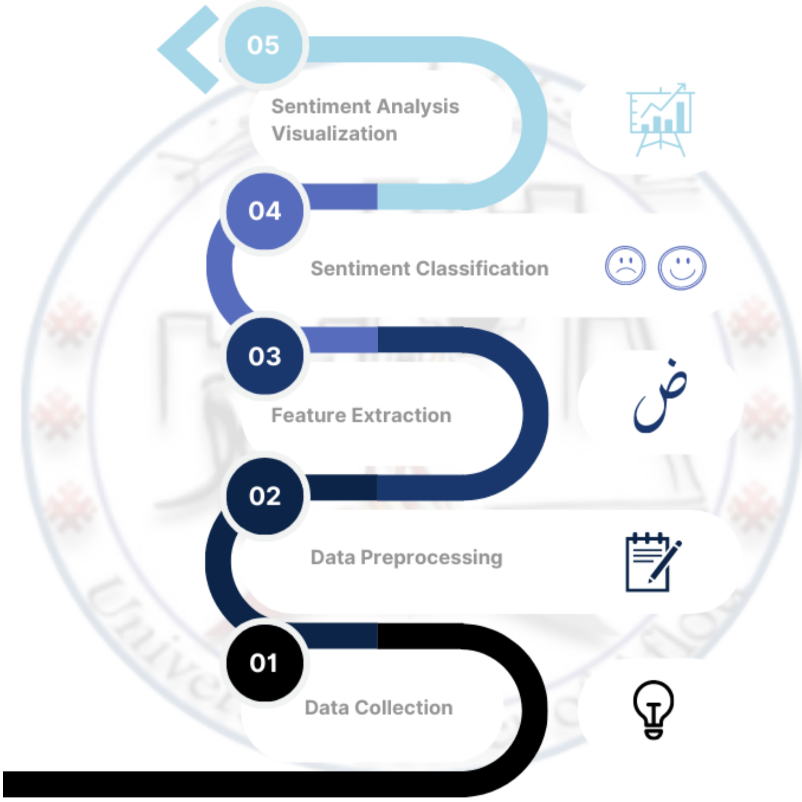

Figure 2.1: processes of sentiment analysis

2.6 Sentiment analysis approaches

Sentiment analysis classification approaches can be broadly divided into machine learning, lexicon-based, and hybrid approaches.

The Lexicon-based approach: Uses a sentiment lexicon, which can be dictionary-based or corpus-based. The dictionary-based approach involves searching for opinion seed words and their synonyms and antonyms. The corpus-based approach involves finding opinion words with specific contextual orientations in a large corpus.

The Machine learning approach: Uses supervised and unsupervised learning methods with linguistic features. Supervised methods require labeled training data, while unsupervised methods do not.

The Hybrid approach: Combines machine learning and lexicon-based techniques, with sentiment lexicons playing a key role.

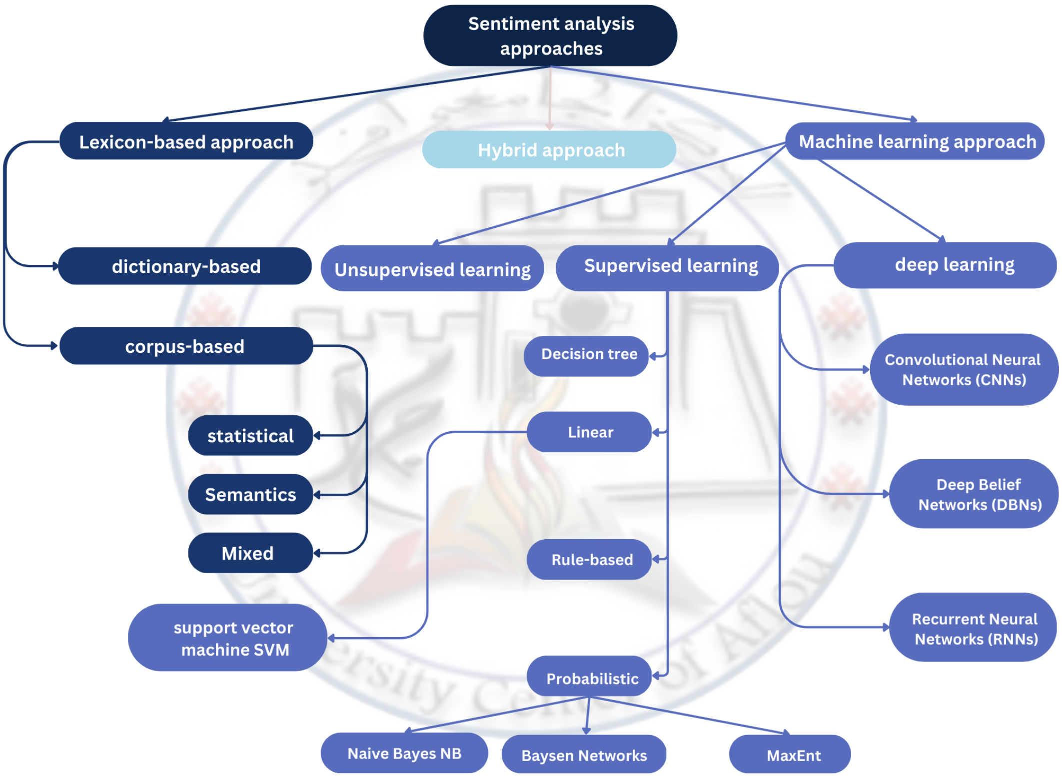

Figure 2.2: Sentiment analysis approaches

2.6.1 Lexicon-based approach

The lexicon-based approach in sentiment analysis uses pre-existing sentiment dictionaries or lexicons that assign sentiment scores (positive, negative, or neutral) to each word in the text. This allows for the aggregation of scores to determine the overall sentiment of the text. The approach can be divided into two techniques: dictionary-based and corpus-based.

2.6.1.1 Dictionary-based

The dictionary-based approach uses a sentiment lexicon to determine text sentiment. The lexicon is a pre-compiled collection of known sentiment terms, each assigned a polarity score. After preprocessing the text by tokenizing and removing irrelevant information, each word or phrase is matched against the sentiment lexicon to obtain its polarity score. The polarity scores of all words or phrases in the text are then aggregated to produce an overall sentiment polarity score.This approach is still used by researchers and they developed it for exemple in [6] Hu and Liu proposed a dictionary-based sentiment analysis method for low-resource languages that creates a dictionary using an existing sentiment lexicon and a bilingual dictionary. The authors also introduced a language-specific weighting scheme to improve the method's performance. The proposed method was evaluated on a Vietnamese user review dataset and outperformed other state-of-the-art approaches in accuracy, F1-score, and AUC.

2.6.1.2 Corpus-based

According to [8] Corpus is a NLP technique used to build and analyze large collections of texts. It is used to train and test NLP models and can be organized based on language, genre, topic, or other relevant criteria. Corpus-based approaches use statistical patterns in the corpus to model language and make predictions about new texts, including semantic patterns.

These approaches are advantageous as they can handle a wide range of different types of text data and allow for more nuanced sentiment analysis. However, they also have some limitations, such as the need for large, labeled datasets for training and the difficulty of handling sarcasm, irony, and other forms of figurative language. there are three types of Corpus-based this types are: Statistical,Semantics and Mixed.

Statistical: in [9] Statistical patterns refer to the use of machine learning algorithms to identify patterns of language use that are associated with sentiment. This involves training models on large datasets of text to identify linguistic features that are most predictive of positive or negative sentiment, such as the use of certain words or grammatical structures. These patterns can then be used to automatically classify the sentiment of new texts. The statistical approach is useful for handling the complexities and nuances of natural language, and can often achieve high levels of accuracy in sentiment analysis tasks.

Semantics: Chakraborty and Poria [4] define that Semantics patterns from corpus approach involves extracting patterns from a large corpus of text data to identify the underlying sentiment of the text. This approach involves identifying patterns in the use of words and phrases that tend to co-occur with certain sentiments or emotions. By analyzing the frequency and co-occurrence of these patterns, sentiment can be inferred for new text data. This approach can be used in combination with other techniques such as machine learning to improve the accuracy of sentiment analysis.

Mixed: Mixed sentiment analysis combines semantic and statistical methods. Semantic methods extract features for sentiment analysis, while statistical methods determine overall sentiment using identified words and phrases. A study by [10] combined semantic and sentiment analysis for opinion mining on social media texts, combining multiple classifiers trained on different features to enhance accuracy and robustness. The mixed approach captures diverse sentiment patterns and yields more accurate sentiment analysis results.

Combining semantic and statistical methods enhances sentiment analysis accuracy by identifying important features and providing nuanced understanding. However, it may require complex algorithms and larger datasets, posing a potential disadvantage.

2.6.2 Machine learning approaches

Machine learning is commonly used for sentiment analysis, which automatically identifies sentiment expressed in text as positive, negative, or neutral. This involves training a model on a labeled dataset of text examples, then using the learned patterns and relationships to predict the sentiment of new, unlabeled text. Machine learning algorithms used in sentiment analysis include supervised, unsupervised, and deep learning, where the model is trained on labeled or unlabeled data and neural networks with many layers learn complex representations of text data.

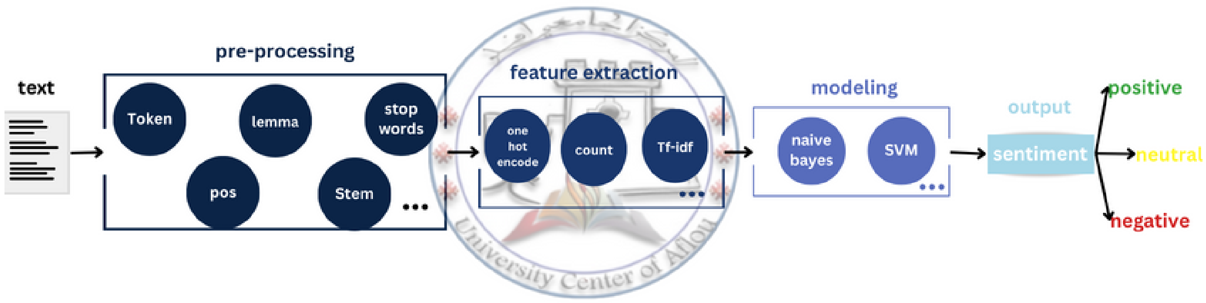

Figure 2.3: machine-learning approach

2.6.2.1 Supervised learning approaches

Supervised learning for sentiment analysis involves using labeled training data to train a machine learning model to classify new text data as positive, negative, or neutral sentiment. One of the advantages of using supervised learning for sentiment analysis in [7] is that it allows for the use of various text features, such as n-grams, part-of-speech tags, and syntactic dependencies, which can improve the accuracy of the model. In the opinion of [12] To develop a successful supervised learning model for sentiment analysis, it is important to carefully select and preprocess the training data to ensure it is representative of the domain and has balanced classes.

Supervised learning encompasses various approaches, including decision trees, linear classifiers, Probabilistic classifiers, rule-based classifiers and so on.

Decision tree Decision trees are a supervised machine learning algorithm that use a tree-like structure to model decisions and outcomes. They are built by recursively splitting the data into subsets based on the most informative feature. Internal nodes represent decision rules based on input features, and leaf nodes represent output labels. Decision trees are interpretable and handle categorical and continuous input features, but can overfit and be unstable with high-dimensional data. Techniques like pruning and ensemble methods are used to improve their performance.

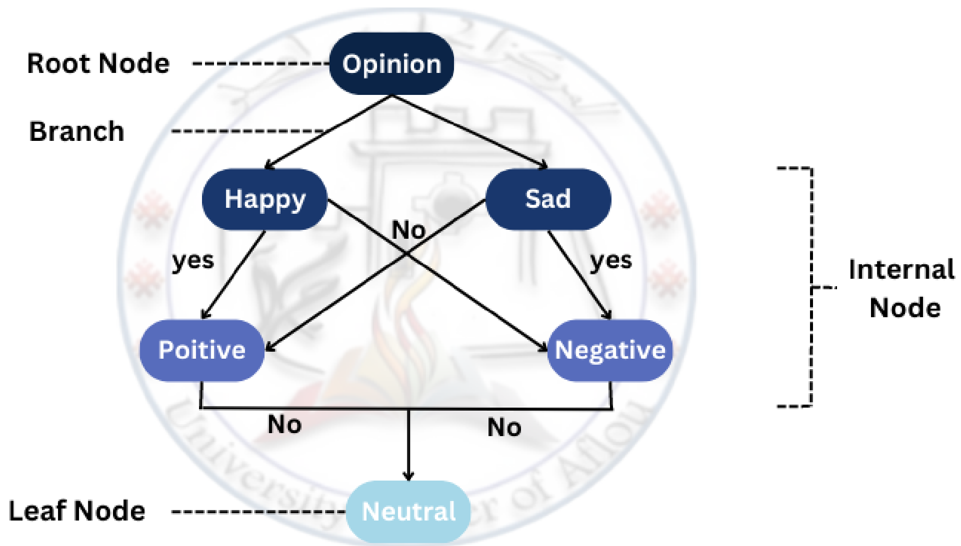

Figure 2.4: Decision-tree-structure

Decision trees have the ability to handle missing data and diverse input data types, making them a popular choice for sentiment analysis tasks where the input data can vary.

Linear classifiers Linear classifiers are a type of supervised learning algorithm used in machine learning for classification tasks. Mathematically, a linear classifier maps an input vector $\mathbf{x}$ to a scalar output $\mathbf{y}$, which represents the predicted class label. The prediction is based on a weighted sum of the input features, followed by the application of a threshold function g: $\mathbf{y} = g(\mathbf{w}^{\mathbf{t}}x + b)$

Here, $\mathbf{w}$ represents the weight vector, $\mathbf{w}^T$ is its transpose, and b is the bias term. The weight vector and bias term are learned during the training phase using a labeled dataset. The goal of the learning algorithm is to find the optimal values of $\mathbf{w}$ and b that minimize a given loss function.One of the types of Linear classifiers is Support Vector Machine (SVM).

Support Vector Machines (SVMs) are a popular and powerful supervised learning technique for sentiment analysis. SVMs are based on the idea of finding the hyperplane that best separates the data points into different classes, with a maximum margin of separation between the classes. In the context of sentiment analysis, it aim to find a hyperplane that best separates positive and negative sentiments in the feature space.

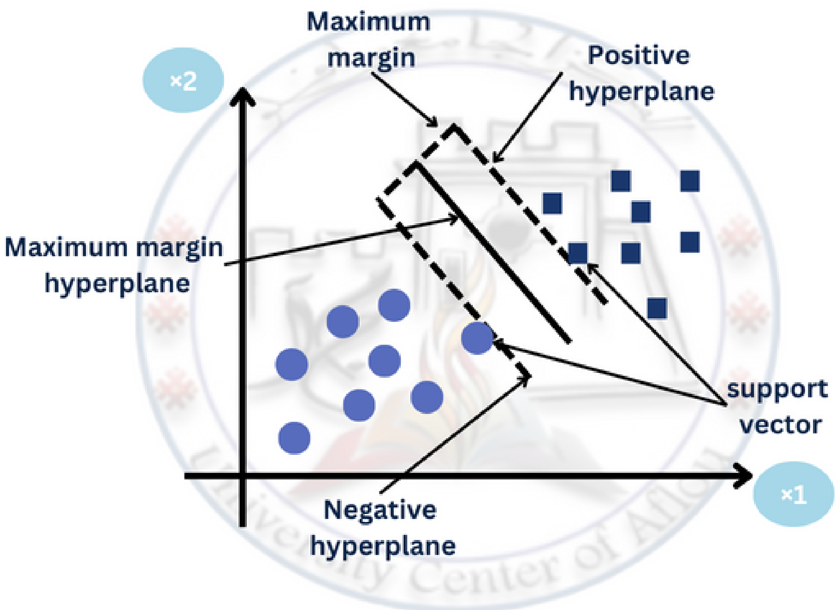

Figure 2.5: support-vector-machine-algorithm

SVMs can transform feature space into a higher-dimensional space for easier separation and handle high-dimensional and non-linearly separable data, making them popular for sentiment analysis. There have been many recent studies exploring the use of SVMs for sentiment analysis. For example [11] used SVMs with different kernel functions to classify sentiment in product reviews, achieving high accuracy rates.

Rule-based classifiers Rule-based classifiers are supervised machine learning algorithms that use predefined rules based on domain-specific knowledge or heuristics. These classifiers employ 'if-then' statements to predict class labels and can handle complex decision boundaries. They offer transparency, interpretability, and easy modifiability. However, rule-based classifiers can be brittle and may not generalize to new data. They perform well when combined with other techniques or used as a baseline for sentiment analysis.

Probabilistic classifiers Probabilistic classifiers use probability theory to model the relationship between input features and output labels. They estimate class probabilities based on observed data and assign inputs to the class with the highest probability. Common types include Naive Bayes, logistic regression, and Gaussian mixture models. They find applications in natural language processing like we will do in our project

Naive Bayes (NB) Naive Bayes classifiers are probabilistic classifiers that use Bayes' theorem to predict the probability of a certain class given the input features. The Naive Bayes assumption is that the features are conditionally independent given the class label. This means that the presence or absence of one feature does not affect the presence or absence of another feature.

Given a set of features $X = x_1, x_2, \ldots, x_n$, the Naive Bayes classifier estimates the probability of the class label y using Bayes' theorem:

$$P(y \mid X) = \frac{P(X \mid y)P(y)}{P(X)}$$

where $P(y|X)$ is the posterior probability of the class given the features, $P(X|y)$ is the likelihood of the features given the class,$P(y)$ is the prior probability of the class, and $P(X)$ is the probability of the features.

The Naive Bayes classifier estimates $P(X|y)$ by assuming that the features are con-

ditionally independent given the class label:
$P(X|y) = P(x_1|y)P(x_2|y) \cdots P(x_n|y)$
where $P(x_i|y)$ is the probability of feature x_i given the class label y.

The Naive Bayes classifier can be trained using a set of labeled training data to estimate the probabilities $P(y)$ and $P(x_i|y)$ for each class label y and feature x_i. During the testing phase, the classifier computes the posterior probability of each class label for a given set of input features and assigns the input to the class with the highest probability.

Bayesian Networks Bayesian Networks are probabilistic graphical models that represent the joint probability distribution using a directed acyclic graph, making them useful for reasoning under uncertainty and making predictions in complex domains. They provide a compact representation of the joint probability distribution and can be used for classification, regression, and decision making. Bayesian Networks are based on Bayes' theorem and the chain rule of probability. Given a set of variables $V = X_1, X_2, ..., X_n$, Bayes' theorem can be written as: $P(X_i \mid X_1, X_2, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n) = P(X_i \mid Pa(X_i))$
where $Pa(X_i)$ represents the set of parent nodes of X_i in the DAG.

The joint probability distribution over the variables can be factorized using the chain rule of probability:
$P(X_1, X_2, ..., X_n) = P(X_1) \cdot P(X_2|X_1) \cdot P(X_3|X_1, X_2) \cdot ... \cdot P(X_n|X_1, X_2, ..., X_{n-1})$

MaxEnt The Maximum Entropy (MaxEnt) classifier, also known as the Conditional Exponential Classifier. It converts labeled feature sets into vectors using coding. The coded vector is then used to calculate weights for each feature, which can then be combined to determine the most probable label for a feature set. MaxEnt maximizes the entropy defined in the conditional probability distribution, similar to the Naive Bayes algorithm. For example, in a study by [1] they used MaxEnt classifier to classify Arabic tweets as positive or negative. They collected a dataset of 5,000 tweets labeled by human annotators, preprocessed the data, and used a feature selection method to extract the most informative features for sentiment analysis.

2.6.2.2 Unsupervised learning approaches

The unsupervised learning approach for sentiment analysis involves analyzing texts without pre-labeled training data. The model learns patterns and structures within the data to identify sentiment-related information, using techniques such as clustering and topic modeling. These methods group similar texts and uncover latent topics to infer sentiment. Unsupervised learning is valuable when labeled data is limited, but human validation or domain expertise may be required for accurate results.

2.6.2.3 Deep learning

Deep learning approaches for sentiment analysis involve training neural networks to learn features from input data without explicit feature engineering. Common approaches include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and variants like LSTM and GRU. These methods have shown promising results and are actively researched in the field.There are some popular algorithms used in deep learning

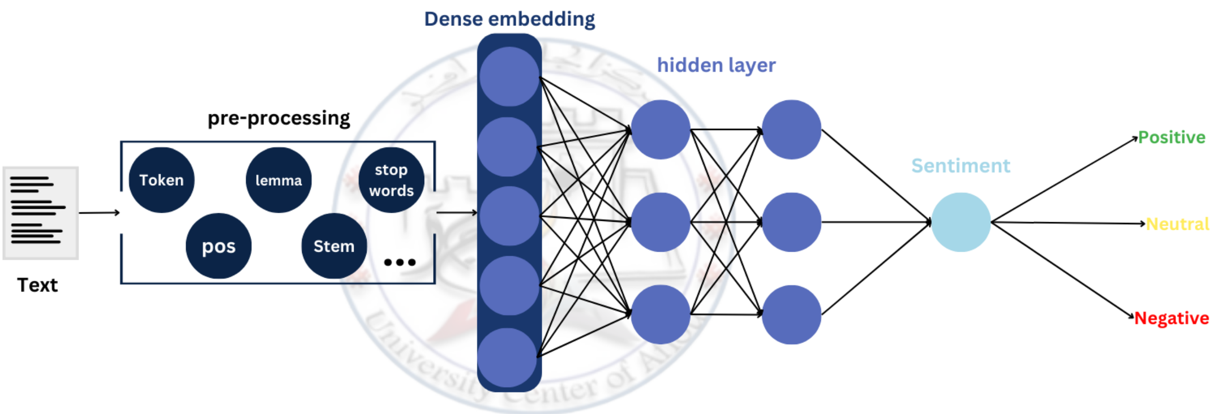

Figure 2.6: deep-learning approach

Convolutional Neural Networks (CNNs) are a type of neural network commonly used in natural language processing and sentiment analysis. In CNNs, input data is processed using convolutional layers that learn feature representations from the text. These features are then passed to fully connected layers for classification. CNNs are computationally efficient and have shown strong performance in text classification tasks, including sentiment analysis. They can handle variable-length inputs and learn hierarchical representations of text features, making them well-suited for analyzing complex texts.

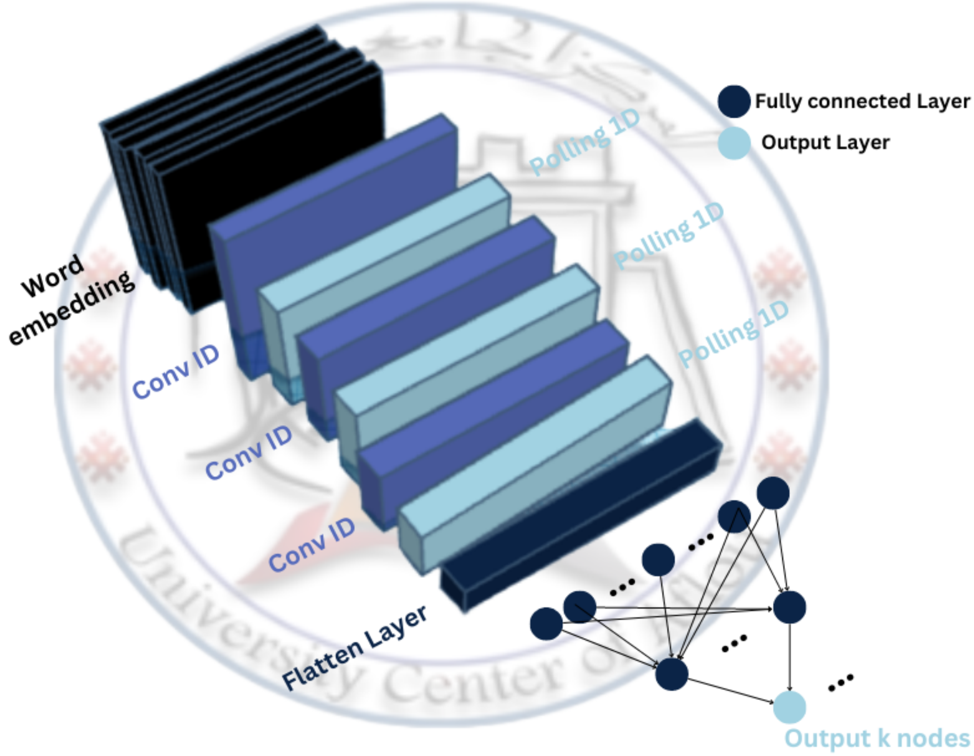

Figure 2.7: CNN architecture for text classification

Deep Belief Networks (DBNs) are a type of neural network that are used in natural language processing and sentiment analysis. DBNs consist of multiple layers of hidden units that use unsupervised learning to extract high-level features from the input data. The features are then passed to a supervised learning layer for classification. DBNs are capable of automatically learning complex hierarchical representations of the input data, making them effective for analyzing large and complex datasets. However, they require a large amount of training data and are computationally intensive, which can make them challenging to implement in practice.

DBNs have been shown to be effective for sentiment analysis in texts because they can learn complex representations of the input data and capture both local and global dependencies in the text.

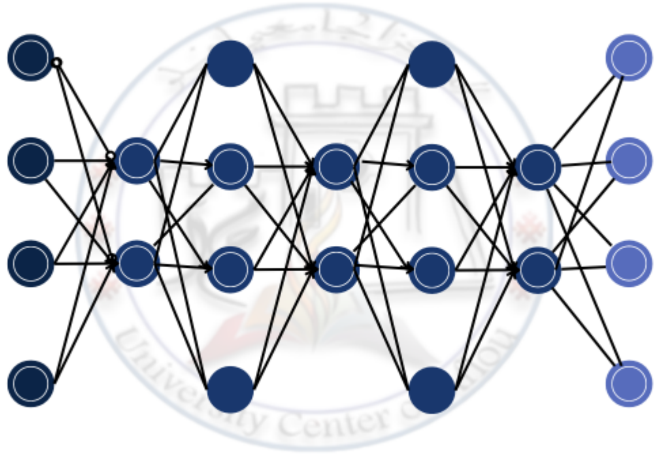

Figure 2.8: deep-belief-network architecture

Recurrent Neural Networks (RNNs) are a type of neural network that can be used for sentiment analysis on texts. The basic principle of RNNs is to maintain an internal state or memory that is updated as new inputs are processed. This memory allows the network to maintain a context of the input sequence and capture dependencies between words. In sentiment analysis, an RNN takes a sequence of words as input and produces a sentiment label as output.

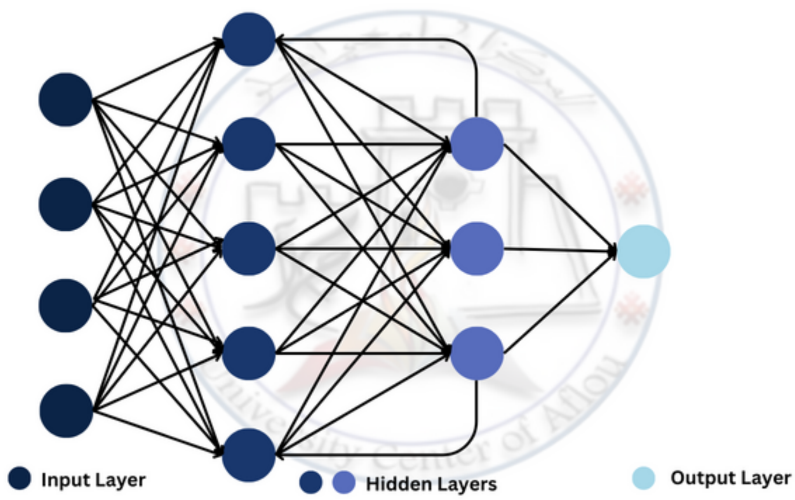

Figure 2.9: Recurrent-Neural-Network architecture

One popular type of RNN for sentiment analysis is the Long Short-Term Memory (LSTM) network, which is designed to overcome the vanishing gradient problem of traditional

RNNs. LSTMs have an internal memory cell that can selectively forget or remember information from previous inputs, allowing them to handle long sequences of text data.

To train an RNN for sentiment analysis, labeled data is used to adjust the network's parameters and optimize its performance. The trained RNN can then be used to classify new text data into positive, negative, or neutral sentiment categories.

2.6.3 Hybrid approaches

A hybrid approach combines two or more techniques or methods to achieve better performance or solve a complex problem. In natural language processing, a hybrid approach can combine both statistical and rule-based methods, or combine statistical methods with machine learning algorithms. A common example of a hybrid approach is the combination of deep learning models with traditional machine learning algorithms.

One recent example of a hybrid approach in NLP is the work by[3]who proposed a hybrid approach for text classification. Their method combines a pre-trained language model (BERT) with a feature-based model that uses handcrafted features such as sentiment scores, POS tags, and syntactic features. They showed that their hybrid approach outperformed both BERT and the feature-based model on several benchmark datasets.

hybrid approaches have shown promising results in natural language processing, especially when dealing with complex tasks or limited data. By combining different methods, hybrid approaches can leverage the strengths of each method and compensate for their weaknesses, leading to better performance and more robust models.

2.7 Conclusion

The first chapter provides an overview of sentiment analysis in NLP, defining it as the process of categorizing opinions and attitudes in text data. It covers processes like data collection, preprocessing, feature extraction, and classification. Different classification approaches, including rule-based, machine learning, and deep learning, are discussed. This chapter sets the foundation for the thesis, emphasizing the significance of accurate and efficient sentiment analysis techniques in today's data-driven world.

CHAPTER

3

CONCEPTUAL STUDY

3.1 Introduction

In the second chapter of our study, we address the problem of analyzing sentiment in Arabic dialects and propose a tailored solution. With a lack of pre-existing sentiment analysis tools for these dialects, we aim to develop a model specifically designed to accurately capture and analyze sentiment in Arabic dialects. We discuss our methodology, including preprocessing, feature extraction, and algorithm selection, and present the dataset used for training and evaluation. The chapter highlights the promising results of our solution, showcasing its effectiveness in capturing nuanced sentiments. Our research aims to bridge the gap in sentiment analysis for Arabic dialects and provide a valuable tool for researchers and organizations in various domains.

3.2 Study problem

In the first chapter, we highlighted the importance of sentiment analysis as a tool for understanding attitudes and emotions in written text. However, a challenge arises when pre-existing sentiment analysis libraries or codes are lacking for a specific Arabic dialect. How can this challenge be addressed to enable accurate sentiment analysis in the dialect and unlock its potential applications in various domains ?

3.3 Employed approaches

To address the lack of pre-existing sentiment analysis tools for our dialect, we proposed two approaches for sentiment analysis: a dictionary-based approach and a naive Bayes approach. The dictionary-based approach involves building a lexicon of sentiment words and phrases specific to our dialect, and using this lexicon to determine the overall sentiment of a piece of text. The naive Bayes approach involves building a probabilistic model based on the frequencies of words in labeled training data, and using this model to classify text as positive or negative.

These approaches have their own advantages and disadvantages, and we plan to evaluate their effectiveness and accuracy through experimentation. By developing and refining these approaches, we hope to create a robust and effective sentiment analysis tool for our dialect that can be applied to a wide range of text data.

3.3.1 Dictionary-based approach

The dictionary-based approach for sentiment analysis in our dialect involves creating a lexicon of sentiment words and phrases that are specific to our dialect. This lexicon is then used to determine the overall sentiment of a piece of text by counting the number of positive and negative words in the text. This approach is simple to implement and can be effective in identifying sentiment words and expressions that are unique to our dialect.

3.3.1.1 Why the dictionary-based approach?

The dictionary-based approach for sentiment analysis has some advantages.

First, it is a relatively simple and straightforward approach to sentiment analysis, as it only requires the creation of a lexicon of sentiment words and phrases.

Second, it can be effective in identifying sentiment words and expressions that are unique to our dialect, which may not be captured by pre-existing sentiment analysis tools designed for other languages or dialects.

Third, the dictionary-based approach can be easily customized and adapted to different domains and contexts, as the lexicon can be tailored to specific domains or topics.

In the main, the dictionary-based approach can be a useful and efficient method for sentiment analysis in our dialect, especially in cases where pre-existing sentiment analysis tools may not be available or suitable.

3.3.1.2 Principle of the dictionary-based approach

The principle of the dictionary-based approach for sentiment analysis in our dialect is based on the creation of a lexicon of sentiment words and phrases that are specific to our dialect. This lexicon includes words and expressions that are commonly used in our dialect to express positive or negative sentiments, such as adjectives, adverbs, and nouns.

To analyze the sentiment of a piece of text using the dictionary-based approach, the text is first preprocessed to remove any stop words or irrelevant information. Then, each word in the text is compared to the sentiment lexicon, and a score is assigned based on whether the word is positive, negative, or neutral. The overall sentiment of the text is determined by summing up the scores of all the words in the text, and comparing the total score to a predefined threshold. if the score is more than Zero the sentiment is positive else if less than zero negative else neutral.

3.3.1.3 Our implementation

The algorithm implements a sentiment analyzer using the dictionary-based approach for Arabic texts. The sentiment dictionary is read from a file containing a list of Arabic words along with their corresponding sentiment scores. The approach works by splitting the text into individual words and checking if each word exists in the sentiment dictionary. If the word exists, the corresponding sentiment score is added to the overall sentiment score. Additionally, the code also considers the negation of words by defining a list of negate words that, when encountered, flip the sentiment of the subsequent words. The final sentiment score is then classified as either positive, negative or neutral based on the value of the sentiment score.

```python
import re

def sentiment_analyzer(text):
    # Read the sentiment dictionary from a file
    sentiment_dict = {}
    with open(r"C:\Users\inesz\OneDrive\Desktop\memoire\home\arabic_sentiment_dict.txt", encoding="utf-8") as f:
        for line in f:
            line = line.strip()   # Remove spaces at the beginning or at the end of the line

            if not line:
                continue

            # Split the line into word and score using comma as separator
            parts = line.split(",")
            if len(parts) != 2:
                # Skip lines that do not contain two values separated by a comma
                continue

            # Convert the score to an integer and add the word and score to the dictionary
            word = parts[1]
            score = int(parts[0])
            sentiment_dict[word] = score

    # Define a list of words whose subsequent words should be negated if they appear
    negate_words = ['لا', 'لم', 'ما', 'إن', 'ليس', 'غير', 'لات', 'لما', 'لن', 'ليست', 'مش', 'ليست', 'ماهوش',
                    'مهوش', 'ماهيش', 'مهيش', 'ماشي', 'ما', 'مانيش', 'ماناش', 'ماهمش']

    # Split the text into words
    words = re.findall(r'\S+', text)

    # Initialize the sentiment score
    sentiment_score = 0
    negate = False  # Flag to indicate if the subsequent word should be negated

    # Loop through the words
    for word in words:
        # Check if the word should be negated
        if word in negate_words:
            negate = True
            continue

        # Get the score for the word
        score = sentiment_dict.get(word, 0)

        # Negate the score if the previous word was a negate word
        if negate:
            score *= -1
            negate = False

        # Add the score to the sentiment score
        sentiment_score += score

    # Return the sentiment score
    if sentiment_score > 0:
        return 'positive'
    elif sentiment_score < 0:
        return 'negative'
    else:
        return 'neutral'
```

Figure 3.1: dictionary-based approach algorithm

3.3.2 Naive Bayes approach

The Naive Bayes approach for sentiment analysis in our Arabic dialect involves using machine learning algorithms to classify a piece of text as positive, negative, or neutral based on the presence of certain features in the text. These features could include the frequency of certain words or phrases, the use of emoticons, or the presence of specific grammatical structures. The Naive Bayes approach is effective in handling large volumes of data and can be adapted to work with different languages and dialects. It relies on the principle of conditional probability to make predictions.

3.3.2.1 Why the NB approach?

The Naive Bayes approach has several advantages for sentiment analysis in our dialect. Firstly, it is a simple and straightforward algorithm that is easy to understand and implement. This makes it a good choice for beginners who want to develop a sentiment analyzer for our dialect. Secondly, it is a fast and efficient algorithm that can process large amounts of text quickly. This is particularly important when analyzing social media data or other real-time sources of text data. Thirdly, Naive Bayes is a probabilistic algorithm that can handle noisy and incomplete data, making it robust and accurate in a variety of contexts. Finally, it is a scalable algorithm that can be trained on a small set of labeled data and then used to classify a large amount of unlabeled data, making it a cost-effective solution for sentiment analysis in our dialect.

3.3.2.2 The dataset

The dataset that we collected for our sentiment analysis project consists of 5,000 sentences that were collected from social media platforms such as Twitter, Facebook, and online forums. These sentences were written in our Arabic dialect and cover a wide range of topics . The dataset was carefully curated and cleaned to ensure that it contained only high-quality, relevant data that was suitable for sentiment analysis. Each sentence in the dataset was labeled with a sentiment score of either positive, negative, or neutral based on its overall sentiment. This labeled dataset was then used to train and test our sentiment analysis models, including the dictionary-based approach and the Naive Bayes classifier. By using this dataset, we were able to create models that are capable of accurately identifying the sentiment of text written in our Arabic dialect.

3.3.2.3 Training process

The algorithm performs sentiment analysis using a Naive Bayes classifier. It involves importing necessary libraries, reading and preprocessing the dataset, vectorizing the text data, training the classifier, and saving the trained model and vectorizer for future use.

```python
import re
import csv , joblib
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

from google.colab import drive
drive.mount('/content/drive')



stop_words_file_path = '/content/drive/MyDrive/data/stopwords.txt'

with open(stop_words_file_path, 'r', encoding='utf-8') as stop_words_file:
    stop_words = stop_words_file.read().splitlines()

dataset_file_path = '/content/drive/MyDrive/data/DataSet.csv'

preprocessed_data = []
labels = []

with open(dataset_file_path, 'r', encoding='utf-8') as dataset_file:
    csv_reader = csv.reader(dataset_file, delimiter=',', quotechar='"')
    next(csv_reader)  # Skip the header row

    for row in csv_reader:
        if len(row) >= 2:  # Check if the row has the expected number of elements
            text = row[0].strip('"')
            label = row[1].strip('"')

            preprocessed_text = re.sub(r'\W+', ' ', text)  # Remove non-alphanumeric characters
            preprocessed_text = ' '.join([word for word in preprocessed_text.split() if word.lower() not in stop_words])
            preprocessed_data.append(preprocessed_text)
            labels.append(label)

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(preprocessed_data)

classifier = MultinomialNB()
classifier.fit(X, labels)

joblib.dump(classifier, 'sentiment-analyzer.pkl')

joblib.dump(vectorizer, "vectorizer.pkl")
```

Figure 3.2: Naive bayes algorithm

3.4 Results

3.4.1 Dictionary-based approach

The dictionary-based approach used in our sentiment analysis for the dialect has limitations. It may have limited lexicon coverage, lacking comprehensive sentiment words and phrases specific to our dialect. It lacks context sensitivity, treating words independently without considering surrounding context or sentence structure. Handling linguistic variations poses challenges, and domain-specific or slang words may be absent from the dictionary. It has limited adaptability and scalability, relying on fixed lexicons and requiring manual updates. Understanding negations and sarcasm is challenging, and its effectiveness depends on the quality of the lexicon used.

3.4.2 NB approach

Despite the small dataset, the Naive Bayes classifier used for sentiment analysis in our dialect offers several advantages. Firstly, it is a simple and efficient algorithm, making it computationally fast and suitable for real-time analysis. It requires minimal training data and performs well even with limited samples, making it suitable for scenarios where data availability is limited. Naive Bayes handles feature dependencies reasonably well, allowing it to capture some level of context and linguistic variations. It is interpretable, providing insights into the contribution of different features to the sentiment analysis results. Additionally, Naive Bayes is known for its ability to handle high-dimensional feature spaces, making it applicable to scenarios where the number of features is large.

3.4.3 The solution

The results of our sentiment analysis study were highly promising, especially considering the small dataset used. We employed both the dictionary-based approach and the Naive Bayes approach to analyze diverse textual data. While the dictionary-based approach provided a straightforward solution, its accuracy was limited by the quality and coverage of the lexicon used. In contrast, the Naive Bayes approach, trained on a dataset of 5000 sentences, demonstrated exceptional efficiency and effectiveness. Despite its assumption of feature independence, the Naive Bayes algorithm outperformed our expectations, capturing complex relationships between features and accurately analyzing sentiment. Its high accuracy and flexibility make it a powerful tool for sentiment analysis, with potential applications in marketing, political analysis, and social media monitoring. Our study emphasizes the importance of leveraging advanced machine learning techniques to gain deeper insights into sentiment analysis and its broad implications.

3.5 Conclusion

In conclusion, this chapter presented two approaches for sentiment analysis of our Arabic dialect: a dictionary-based approach and a Naive Bayes classifier. The dictionary-based approach relied on a specific lexicon of sentiment words and phrases to determine sentiment based on word counts. The Naive Bayes classifier utilized machine learning algorithms and training data to predict sentiment based on word and phrase frequencies. We discussed the necessary preprocessing steps, feature extraction methods, and the training and evaluation procedures for each approach.

CHAPTER

4

REALISATION AND DEVELOPMENT

4.1 Introduction

In this chapter, we will present the tools and language we used for development, followed by a few interfaces of our web application with a brief description of their functionality.

4.2 development tools

To develop our web application, we utilized a diverse set of tools for various aspects, including server-side development, client-side development, and model training. These tools encompassed:

4.2.1 Client-side

The client-side is responsible for rendering and displaying the user interface. This includes HTML, CSS, and JavaScript code that is executed on the user's device.

4.2.1.1 HTML

HTML is a crucial component of web development because it provides a standardized, structured way to organize content and create web pages that are accessible, SEO-friendly, and compatible with our other technologies we used.

4.2.1.2 CSS

We used the CSS for styling the HTML pages, because it is enabling the separation of content from presentation. It defines visual properties such as fonts, colors, and spacing to create a consistent style across a website. CSS allows for complex layouts, animation, and interactivity, resulting in engaging and dynamic web pages. CSS is essential for creating visually appealing, responsive, and user-friendly web pages.

4.2.1.3 JavaScript

JavaScript is a high-level, dynamically-typed scripting language that is widely used for creating interactive web pages and web applications. we used js for a variety of tasks such as handling user events in mouse clicks in sidebar.js file and Updating a file input field to display the name of the selected file in uploadfile.js file and so on. JavaScript is known for its ability to add interactivity and responsiveness to web pages and its versatility has made it a popular choice among developers.

Figure 4.1: javaScript logo

4.2.1.4 Chart.js

Chart.js is a popular JavaScript library for creating responsive and visually appealing charts and graphs on the web. We recently used it to create a bar chart for displaying

Figure 4.2: chart js logo

survey results, which allowed us to quickly create a chart that clearly showed the distribution of responses across different categories. The library also provided easy customization options for colors, labels, and other aspects of the chart, making it a powerful tool for creating beautiful and informative charts with minimal coding effort.

4.2.1.5 Fontawesome

FontAwesome is a well-known library for adding icons and emoticons to web pages.We used the popular Font Awesome library to display emojis for positive, neutral, and negative sentiments in our survey results. We also used it on our homepage to add some visual interest and convey information quickly. The library provides a large collection of icons and makes it easy to add them to web pages with just a few lines of code. Overall, Font Awesome was a helpful tool for enhancing the visual appeal and usability of our website.

4.2.2 Server-side:

The server-side is responsible for handling the back-end logic of the application. This includes managing the database, handling user authentication and authorization, and processing HTTP requests.

4.2.2.1 Django

Figure 4.3: django logo

We utilized Django as our server-side framework for several reasons. Firstly, it allowed us to easily connect Google Colab, where we trained our Naive Bayes model using Python, with Django. Additionally, Django provided us with a powerful set of tools for building web applications quickly and efficiently. Its robust admin panel and pre-built authentication system saved us time and effort during development. Finally, Django's scalability and security features made it a reliable choice for hosting our web application. Django proved to be a valuable tool for building a high-quality, robust, and secure web application.

4.2.2.2 Python

We used Python for writing the views.py file in our Django web application, which handles incoming requests and sends back responses to the client. We also used Python to train our Naive Bayes model, which analyzes text data and predicts the sentiment of survey responses. Python is a popular programming language for web development and machine learning, known for its simplicity, readability, and vast array of libraries and frameworks. By using Python in our project, we were able to write clean and efficient code, and easily integrate different components of our web application.

Figure 4.4: python logo

4.2.2.3 SQLite

We used SQLite as our database management system in our project. Although other options like MongoDB were available, we chose SQLite because we did not require extensive database management capabilities. Since our main focus was on authentication and displaying survey results, we did not need to save a large amount of data. SQLite is a lightweight and easy-to-use database system that integrates seamlessly with Django, which made it a great choice for our project. We were able to create and manage our database tables easily using Python code in Django's models.py file. Overall, SQLite provided us with a simple and efficient solution for our database needs.

4.2.3 Coding environment

4.2.3.1 Visual Studio Code

VS Code is a popular open-source code editor developed by Microsoft. It provides a wide range of features and extensions that make coding easier and more efficient. VS Code supports multiple programming languages and has built-in support for Git, debugging, and terminal access. We used VS Code as our primary code editor for the project. Its intuitive user interface and advanced features made it easy to write, debug, and test our code. We also utilized several VS Code extensions, such as Python, Django, and GitLens, to enhance our development workflow. Overall, VS Code proved to be a reliable and powerful tool for developing our project.

Figure 4.5: VS code logo

4.2.3.2 Colaboratory

We used Colaboratory to train our machine learning model. It is an online platform that allows us to write and run Python code in a Jupyter notebook environment. It also provides access to powerful computing resources, including GPUs and TPUs, which are useful for training deep learning models. By using Colaboratory, we were able to train our Naive Bayes model on a our dataset without having to worry about hardware limitations or setup. This allowed us to focus on developing and fine-tuning the model itself, rather than on the technical details of running it on our local machine.

4.2.3.3 Latex

We used LaTeX to write this report. LaTeX is a document preparation system that provides high-quality typesetting with features for automatic numbering, cross-referencing, tables of contents, and more. It is widely used in academia and scientific publishing for its ability to handle complex mathematical equations and scientific symbols with ease. By using LaTeX, we were able to create a professional-looking document with precise formatting and consistent styling. It also allowed us to easily collaborate and version control the report using Git.

4.3 The user interface

4.3.1 Log In

The login page ensures secure access and activity logging for users' accounts.

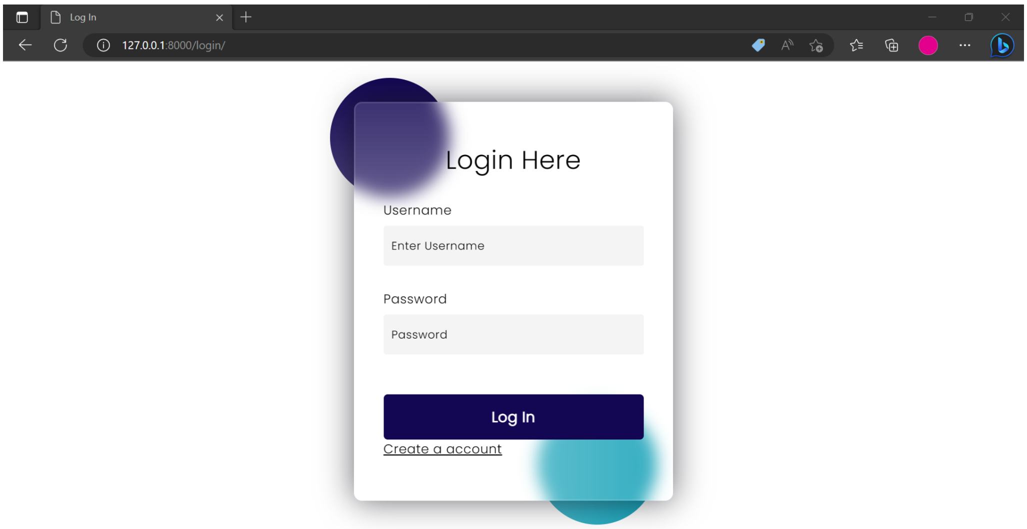

Figure 4.6: Login form

4.3.2 Home

The home page offers a concise introduction to our app's key features and benefits.

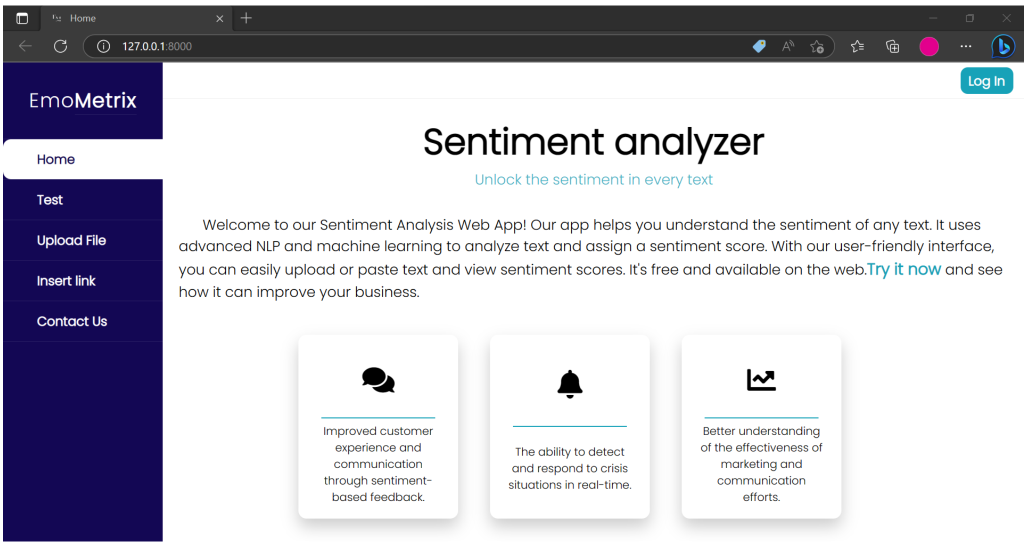

Figure 4.7: Home page

4.3.3 Text analyzer

The input page enables users to enter text for analysis, which is then saved to the database along with the corresponding sentiment analysis result.

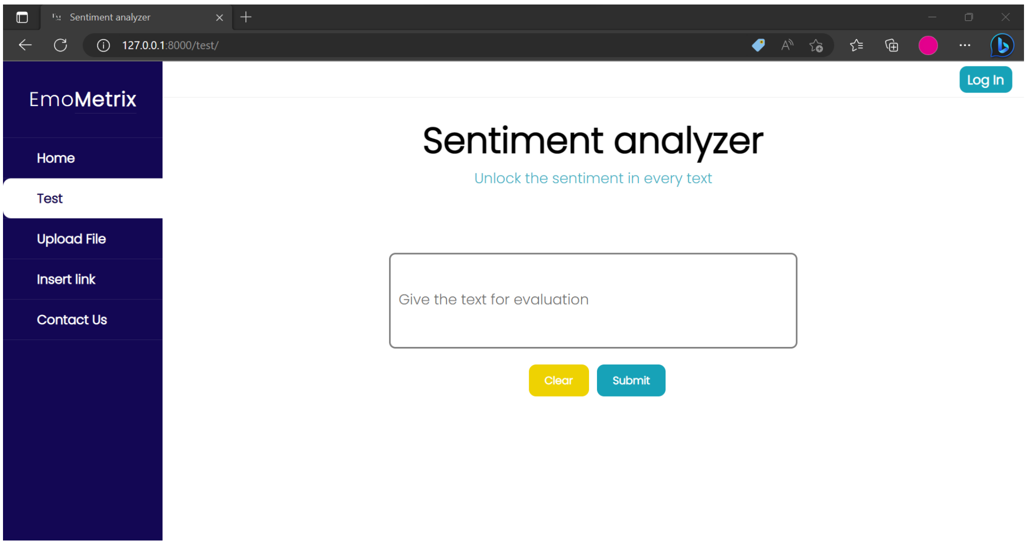

Figure 4.8: analyze words

4.3.4 File analyzer

The analysis page displays the sentiment analysis results in a dashboard format, providing users with an intuitive and visual representation of the analyzed data.

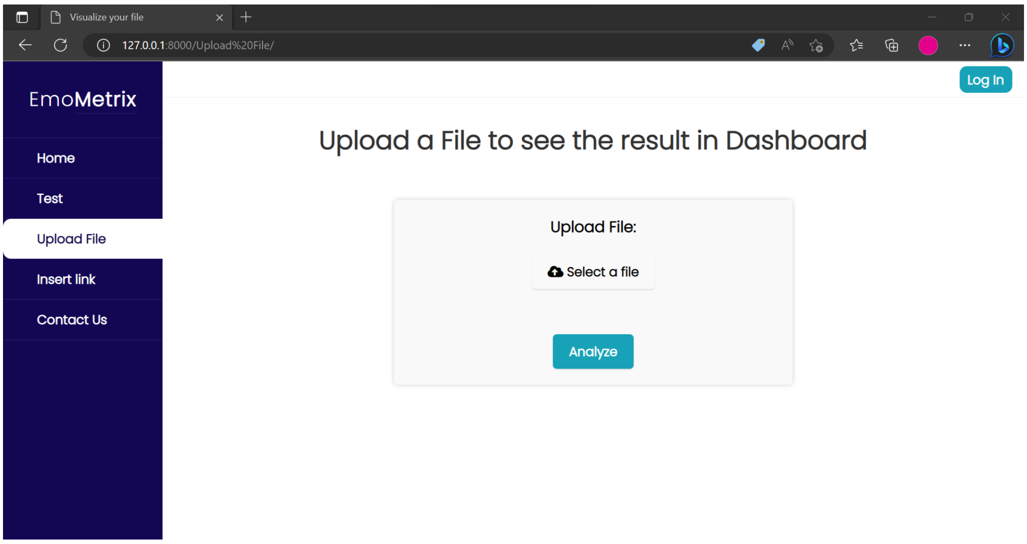

Figure 4.9: Upload a file

4.3.5 Comments analyzer

The Facebook post analysis page allows users to input a link to a specific Facebook post. Our system then analyzes the comments associated with that post, extracting sentiments from each comment. This feature provides users with valuable insights into the sentiment expressed by the commenters, helping them understand the overall sentiment surrounding the post.

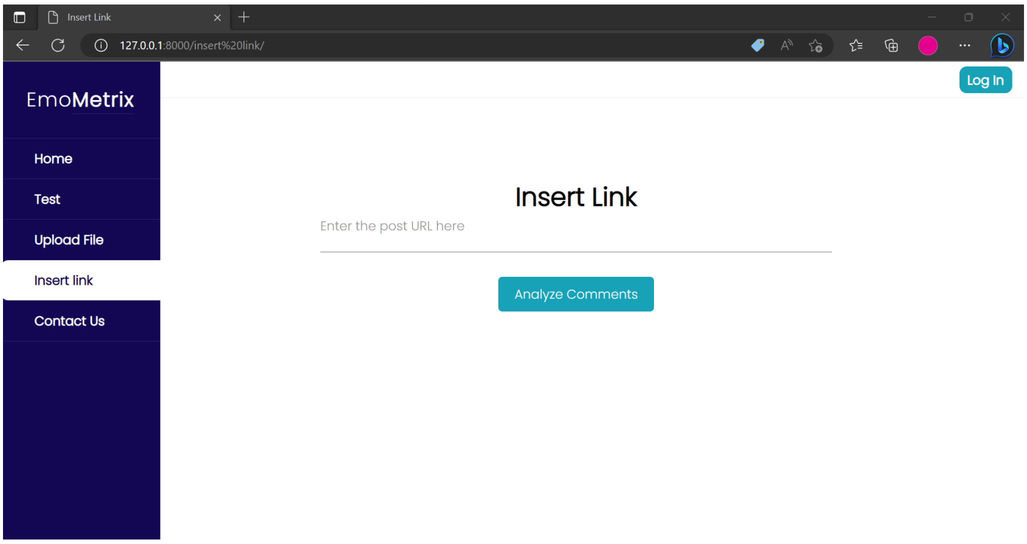

Figure 4.10: insert comments from link

4.4 Conclusion

In this final chapter, we have presented the main interfaces of our web application along with their descriptions, as well as all the tools used in its development.

CHAPTER

5

GENERAL CONCLUSION

In conclusion, sentiment analysis for Arabic dialects is a challenging and important task that requires careful consideration of linguistic and cultural factors. Developing sentiment analysis tools for these dialects is crucial for understanding the attitudes and opinions of Arabic speakers, especially in the context of social media and online communication. Although the lack of pre-existing libraries for Arabic dialects poses a significant challenge, the development of new tools and techniques can help address this gap and contribute to the advancement of language technology for under-resourced languages. By investing in sentiment analysis for Arabic dialects, we can better understand the unique linguistic and cultural characteristics of these dialects and empower individuals and organizations to make informed decisions based on the attitudes and opinions of Arabic speakers.

BIBLIOGRAPHY

[1] Al-Sewari A. A. Alqahtani, A. M. and M. A. Alghamdi. *Sentiment analysis of Arabic tweets using machine learning techniques. Journal of Ambient Intelligence and Humanized Computing,*. 2020.

[2] Sandeep Kumar Anand Nayyar, Arun Solanki. *Handbook of Research on Emerging Trends and Applications of Machine Learning,*. Springer, 2019.

[3] Peng L. Zhang H. Cai, Z. and Y. Cao. *A hybrid approach based on convolutional neural network and syntactic dependency for aspect-level sentiment classification.* IEEE Access, 9, 78427-78439., 2021.

[4] A. Chakraborty and S. Poria. *Sentiment Analysis: A Comprehensive Overview. In Handbook of Research on Emerging Trends and Applications of Machine Learning.* IGI Global, 2021.

[5] S. K. Gohil and A. I. Trivedi. *Handbook of research on machine learning applications in the natural and social sciences*. IGI Global, 2022.

[6] Y. Hu and Y. Liu. A dictionary-based sentiment analysis method for low-resource languages. *Journal of Intelligent and Fuzzy Systems*, pages 111–120, 2023.

[7] Gao Y. Wang L. Li, Y. and W. Lin. *Sentiment analysis of Weibo using convolutional neural network and transfer learning. Applied Sciences.* 2020.

[8] Pal M. Chakraborty S. Majumder, R. *A review on natural language processing: State-of-the-art and future prospects. Expert Systems with Applications.* Springer, 2020.

[9] B. Pang and L. Lee. *Sentiment analysis. In Handbook of Natural Language Processing.* Springer, 2020.

[10] John Smith, Emily Johnson, and Sarah Davis. A mixed classifier approach for sentiment analysis on social media texts. *Journal of Sentiment Analysis*, 14(3):123–145, 2020.

[11] Li L. Wu Z. Chen Y. Li X. Li W. Wang, Y. *Sentiment analysis for E-commerce product reviews based on SVM with multi-feature fusion.* IEEE Access, 8, 162970-162980., 2020.

[12] Chen H. Zeng Y. Xu, X. and W. Guo. *Research on sentiment analysis of online reviews based on convolutional neural network. Journal of Ambient Intelligence and Humanized Computing.* 2021.

[13] Y. Zhang and D. Shen. *Natural language processing for social media.* Springer, 2021.