République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Université de 8 Mai 1945 – Guelma -
Faculté des Mathématiques d'Informatique et des Sciences de la matière
Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique
Option : Systèmes Informatiques

Thème :

Reconnaissance des dialectes algériens avec des modèles de langage fine-tunés

Présenté par : ROUABHIA Nor EL Houda

Membres du jury

Encadreur : Pr. KOUAHLA Mohamed Nadjib

Président : Dr. KHEBIZI Ali

Examinateur : Dr. BOUGHAREB Djalila

Juin 2025

Résumé

L'analyse des sentiments a connu un regain d'intérêt considérable avec l'essor de l'intelligence artificielle, notamment pour l'étude des opinions exprimées sur les réseaux sociaux. Ce mémoire vise à répondre aux défis liés à la faible disponibilité de données annotées en dialecte algérien et à améliorer la précision des modèles dans la détection des sentiments. Pour cela, un fine-tuning a été appliqué à six modèles de langage pré-entraînés (AraBERT, CAMeLBERT, QARiB, mBERT, XLM, DistilBERT) pour une classification binaire (positif/négatif). Le modèle QARiB a obtenu les meilleurs résultats avec une accuracy de 91,1 %. Une évaluation croisée sur des corpus en dialectes marocain et tunisien a permis d'évaluer la capacité de généralisation des modèles. Ce travail apporte une contribution notable au traitement automatique du dialecte algérien et ouvre la voie à de futures recherches sur les modèles multi-dialectaux.

Mots clés : Dialecte algérien, Analyse de sentiments, Fine-tuning, Modèles de langage pré-entraînés (LLMs).

Abstract

Sentiment analysis has experienced a significant resurgence of interest with the rise of artificial intelligence, particularly in studying opinions expressed on social media. This thesis aims to address the challenges related to the limited availability of annotated data in Algerian dialect and to improve model accuracy in sentiment detection. To this end, fine-tuning was applied to six pre-trained language models (AraBERT, CAMeLBERT, QARiB, mBERT, XLM, DistilBERT) for binary classification (positive/negative). The QARiB model achieved the best results with an accuracy of 91.1%. A cross-evaluation on Moroccan and Tunisian dialect corpora was conducted to assess the models' generalization ability. This work makes a significant contribution to the automatic processing of the Algerian dialect and paves the way for future research on multi-dialectal models.

Keywords: Algerian dialect, Sentiment analysis, Fine-tuning, Pre-trained language models (LLMs).

الملخص

تحليل المشاعر شهد انتعاشًا كبيرًا مع تطور تقنيات الذكاء الاصطناعي، خاصة فيما يتعلق بدراسة الآراء المعبر عنها على وسائل التواصل الاجتماعي.

يهدف هذا البحث إلى معالجة التحديات المرتبطة بندرة البيانات المشروحة الخاصة باللهجة الجزائرية، وتحسين دقة النماذج في كشف المشاعر. لهذا الغرض، تم تطبيق تقنية التعلّم الدقيق (Fine-tuning) على ستة نماذج لغوية مُدربة مسبقًا (AraBERT، CAMeLBERT، QARiB، mBERT، XLM، DistilBERT) من أجل تصنيف ثنائي (إيجابي/سلبي). وقد حقق نموذج QARiB أفضل النتائج بدقة بلغت 91.1٪. كما تم إجراء تقييم تقاطعي على مجموعات بيانات بلهجات مغاربية (المغربية والتونسية) لاختبار قدرة النماذج على التعميم. يُعد هذا العمل مساهمة مهمة في مجال المعالجة الآلية للهجة الجزائرية، ويفتح آفاقًا واعدة لأبحاث مستقبلية حول النماذج متعددة اللهجات.

الكلمات المفتاحية: اللهجة الجزائرية، تحليل المشاعر، الضبط الدقيق(Fine-tuning) ، نماذج اللغة المُدرّبة مسبقًا(LLMs)

Remerciements

Avant tout, je tiens à exprimer ma profonde gratitude à **Allah** pour m'avoir accordé la force, la santé, la patience et la persévérance tout au long de ce parcours académique.

Je souhaite ensuite adresser mes remerciements les plus sincères à mon encadreur, **Monsieur Kouahla Mohamed Nadjib**, pour sa disponibilité, ses conseils avisés, sa bienveillance et son accompagnement tout au long de ce projet. Un grand merci également à **Younes** pour son aide précieuse et ses orientations durant les périodes d'absence de mon encadrant.

J'adresse aussi ma gratitude aux **membres du jury** pour le temps qu'ils ont consacré à l'évaluation de ce travail, ainsi qu'à tous nos **enseignants**, qui ont su, tout au long de notre parcours universitaire, nous transmettre leur savoir avec passion.

Je remercie du fond du cœur mes **chers grands-parents**, ma grand-mère et mon grand-père, pour leur amour, leurs encouragements constants et les valeurs qu'ils m'ont transmises : l'humilité, l'effort et la persévérance.

À mes **parents**, mes piliers, merci pour vos sacrifices, votre confiance indéfectible et votre soutien sans faille. Votre amour m'a toujours porté et motivé.

À mes **sœurs**, merci pour votre présence bienveillante, vos encouragements et votre soutien dans les moments difficiles comme dans les moments de joie.

Je n'oublie pas mes **amies fidèles**, qui m'ont accompagnée, soutenue et encouragée tout au long de cette aventure, par leur écoute et leur positivité.

À toutes et à tous, **merci du fond du cœur**.

Table des matières

Liste des figures

Liste des Tableaux

Liste des acronymes

BERT	Bidirectional Encoder Representations from Transformers
BOW	Bag of Words
CNN	Convolutional Neural Networks
DL	Deep Learning
DT	Decision Tree
FN	False Negative
GPT	Generative Pre-trained Transformer
K-NN	K-Nearest Neighbour
LLM	Large Language Model
LR	Logistic Regression
LSTM	Long Short-Term Memory
MADAR	Multi-Arabic Dialect Applications and Resources
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSA	Modern Standard Arabic
NB	Naïve Bayes
NER	Named Entity Recognition
NLP	Natural Language Processing
NMF	Non-negative Matrix Factorization
NMT	Neural Machine Translation
PADIC	Parallel Arabic Dialect Corpus
QA	Question Answer
QCRI	Qatar Computing Research Institute
RF	Random Forest

RNN Recurrent Neural Network

SVM Support Vector Machine

TF-IDF Term Frequency-Inverse Document Frequency

TN True Negative

TP True Positive

Introduction générale

Le traitement du langage naturel constitue l'un des domaines les plus dynamiques de l'intelligence artificielle. Il vise à permettre aux machines de comprendre, générer et interagir avec le langage humain. Ces dernières années, ses applications se sont largement étendues, allant de la traduction automatique à la génération de texte, en passant par la reconnaissance d'entités nommées, ou encore l'analyse de sentiments. Cette dernière est aujourd'hui largement exploitée dans des domaines tels que le marketing, la politique, la santé et le service client, afin de mieux comprendre les opinions et les comportements des utilisateurs.

L'émergence des Large Language Models, tels que BERT, GPT ou XLM-R, a profondément transformé le domaine du NLP. Grâce à leur entraînement sur d'énormes corpus textuels multilingues, ces modèles ont atteint des performances remarquables sur de nombreuses tâches, y compris l'analyse de sentiments. Leur architecture basée sur les Transformers permet de capturer des relations contextuelles profondes et de s'adapter à divers langages via le fine-tuning. La majorité des modèles de traitement du langage existants sont entraînés sur l'arabe standard moderne (MSA), une forme formelle et standardisée utilisée principalement dans les médias et l'écrit. Toutefois, le MSA est rarement employé dans les échanges quotidiens, en particulier sur les réseaux sociaux, où les utilisateurs s'expriment majoritairement dans leurs dialectes locaux. Cela rend nécessaire le développement de modèles capables de traiter la variabilité et la complexité des dialectes arabes.

L'un des dialectes les plus répandus, mais aussi les moins étudiés, est le dialecte algérien, parlé par plus de 40 millions de personnes en Algérie et dans les pays voisins. Ce dialecte se caractérise par une forte hétérogénéité régionale et une influence notable de plusieurs langues, notamment le berbère, le français, l'espagnol et le turc. Il présente également un phénomène de code-switching fréquent, où les locuteurs alternent entre plusieurs langues ou registres au sein d'une même phrase.

Ces spécificités rendent le dialecte algérien particulièrement difficile à modéliser. À cela s'ajoute une autre contrainte majeure : le manque de ressources linguistiques de qualité. La plupart des corpus disponibles sont de taille réduite, bruités, ou spécifiques à certains domaines, ce qui limite l'entraînement de modèles robustes et généralisables. Par ailleurs, les travaux

antérieurs reposent encore souvent sur des approches classiques ou des réseaux neuronaux peu profonds, n'exploitant pas pleinement les récents progrès en matière de modèles de langage pré-entraînés (LLMs).

C'est dans ce contexte que s'inscrit notre travail, dont l'objectif est d'adapter des modèles de langage pré-entraînés (LLMs) à la tâche d'analyse de sentiments en dialecte algérien. Pour cela, nous avons constitué un corpus annoté binaire, appliqué un ensemble d'étapes de prétraitement linguistique, puis effectué le fine-tuning des différents modèles sélectionnés (pré-entraînés multilingues et spécialisés en arabe) et comparé leurs performances sur notre jeu de données. En complément, nous avons testé la capacité de ces modèles à généraliser sur d'autres dialectes arabes proches, à savoir le tunisien et le marocain.

À travers ce travail, nous cherchons à répondre aux questions de recherche suivantes :

- Comment peut-on adapter (fine-tuner) efficacement des modèles de langage pré-entraînés pour la reconnaissance du dialecte algérien ?
- Quels sont les modèles pré-entraînés les plus adaptés pour être fine-tunés sur des données annotées en dialecte algérien dans le cadre de l'analyse de sentiments ?
- Les modèles entraînés exclusivement sur des données en dialecte algérien peuvent-ils être généralisés efficacement à d'autres variétés dialectales arabes ?

Ce mémoire est structuré en trois chapitres principaux :

- Le premier chapitre présente les fondements théoriques de notre étude. Il débute par une description générale de la langue arabe et de ses variétés, en mettant un accent particulier sur le dialecte algérien. Ensuite, il expose les approches traditionnelles et modernes utilisées pour l'analyse de sentiments, ainsi qu'une revue des travaux antérieurs portant sur le dialecte algérien dans différentes tâches de NLP.
- Le deuxième chapitre est consacré à la description de l'architecture générale et détaille toutes les étapes de notre méthodologie.
- Le dernier chapitre présente l'environnement de développement, les bibliothèques utilisées, les principales étapes de l'implémentation, ainsi que les résultats obtenus. Il inclut plusieurs expérimentations approfondies, notamment des évaluations complémentaires sur des dialectes arabes proches (tunisien et marocain), réalisées avant et après prétraitement, afin de tester la capacité de généralisation des modèles entraînés sur le dialecte algérien.

Enfin, les trois chapitres se concluent par un résumé général qui récapitule les points clés de ce mémoire et propose quelques pistes pour des travaux futurs.

Chapitre 1 : Etat de l’art

1 Introduction

Le traitement du langage naturel (NLP) a connu une avancée remarquable avec l'émergence des modèles de langage pré-entraînés appelés LLMs (Large Language Models). Cependant, les dialectes arabes, et en particulier le dialecte algérien, restent encore peu explorés en raison de leur absence de standardisation et du manque de ressources adaptées.

Ce chapitre commence par une présentation des principales formes de la langue arabe, en mettant l'accent sur le dialecte algérien et les défis qu'il pose. Il aborde ensuite les principales tâches NLP associées, notamment l'analyse de sentiment, ainsi que les méthodes d'analyse allant des approches basées sur des lexiques aux techniques avancées de deep learning et les LLM. Enfin, une synthèse des études récentes liées au dialecte algérien dans diverses tâches NLP est proposée.

2 Langue arabe

L'arabe est une langue parlée par environ 420 millions de personnes à travers le monde. C'est la langue officielle de 22 pays, et elle se divise en trois catégories principales [1].

2.1 Arabe classique

L'arabe classique, utilisé principalement dans le Coran et les premiers écrits littéraires de la péninsule arabique, demeure la langue de référence pour les études religieuses, historiques et culturelles. Il sert aussi de base à la littérature arabe contemporaine [2].

2.2 Arabe standard moderne

L'arabe standard moderne (Alfus'ha) est une version modernisée de l'arabe classique, utilisée comme langue officielle dans tous les pays arabes. Ce n'est pas une langue maternelle, mais elle est apprise comme langue seconde à l'école. Elle est utilisée dans les médias, les discours formels, les pratiques religieuses et la presse. Elle permet une communication commune entre les arabophones de différentes régions [3].

2.3 Dialecte arabe

Le terme arabe dialectal désigne les formes parlées de l'arabe, qui diffèrent de l'arabe classique et de l'arabe standard moderne. Ces dialectes varient considérablement d'une région à l'autre, ce qui peut rendre difficile la compréhension entre locuteurs de différentes régions du monde

arabe. Les différences se manifestent tant dans la phonologie que dans la syntaxe, et ces variations sont souvent influencées par des langues européennes telles que le français, l'espagnol, et l'italien. La plupart des études décrivent les dialectes arabes selon une dichotomie est-ouest :

- Les dialectes du Moyen-Orient (péninsule arabique, Syrie, Irak, Egypte, etc.).
- Les dialectes du Maghreb (Algérie, Tunisie, Maroc, Libye, etc.) [4].

Afin d'illustrer la diversité des dialectes arabes, le tableau 1.1 ci-dessous rassemble différentes façons d'exprimer la phrase « Comment vas-tu ? » dans plusieurs dialectes.

Dialecte	**Écriture arabe**	**Transcription latine**
Arabe standard moderne	كَيْفَ حالُك؟	Kayfa haluka ?
Dialecte algérien	كيفاش راك؟	Kifach rak ?
Darija marocaine	كيداير؟	Kidayr ?
Dialecte tunisien	شنوّة حالك؟	Chnoua ḥalek ?
Dialecte libyen	شن حالك؟	Shin ḥalek ?
Arabe égyptien	عامل إيه؟	3amel eih
Dialecte syrien	كيفك؟	Kifak ?

Tableau 1. 1. Exemple de différentes façons de dialecte

2.3.1 Dialecte Algérien

Le dialecte algérien est une variété linguistique hybride influencée par l'arabe standard, le tamazight, ainsi que plusieurs langues étrangères comme le français, le turc et l'espagnol. Il se caractérise par une grande variabilité d'une région à l'autre et peut être écrit dans des alphabets arabes, latins ou un mélange des deux [5]. Ce dialecte ne suit pas une syntaxe ou une grammaire spécifique, et un même mot peut avoir plusieurs significations selon la région en Algérie. Cette absence de standardisation grammaticale, combinée à une forte variation régionale, rend son traitement par des modèles pré-entraînés (tels que ceux adaptés à l'arabe standard ou aux modèles multilingues) particulièrement difficiles. Bien que largement parlé au quotidien par la majorité des Algériens, il n'est pas utilisé dans les écoles, les journaux ou la télévision, où l'arabe standard et le français prédominent. Il est cependant plus fréquemment entendu dans les chansons, ainsi que dans les foyers et dans la rue [4].

2.3.2 Défis de Dialecte Algérien

Le traitement du dialecte algérien présente plusieurs défis majeurs en raison de sa nature non standardisée et de sa forte variabilité linguistique. Parmi les principaux obstacles, on peut citer :

- **Alternance codique (code-switching) :** Les locuteurs algériens alternent entre plusieurs langues ou dialectes au sein d'un même discours, ce qui complique l'analyse syntaxique et sémantique du texte. Par exemple, l'expression « اعطيني solution » combine l'arabe et le français pour signifier « donne-moi une solution ».

- **Translittération entre alphabets :** Le dialecte algérien est fréquemment écrit à l'aide de l'alphabet latin, dans une forme communément appelée arabizi. Cette écriture informelle est largement adoptée par les locuteurs, notamment sur Internet et les réseaux sociaux. Par exemple, « khdma kbira » correspond à « خَدْمَة كَبِيرَة » en arabe, signifiant « grand travail ». Inversement, certains mots d'origine étrangère peuvent être adaptés en alphabet arabe, comme « بَاي بَاي » pour « bye bye ».

- **Utilisation de chiffres pour représenter des lettres :** Les internautes algériens emploient parfois des chiffres pour remplacer des lettres arabes absentes de l'alphabet latin. Par exemple, le chiffre « 7 » est utilisé pour représenter la lettre « ح » et le « 9 » pour « ق ».
- **Variations régionales :** il présente des différences significatives selon les régions. Par exemple, le mot « femme » se traduit par « مَرْأَة » (mra) à l'est du pays, tandis qu'à l'ouest, on utilise « شِيْرَة » (shiira) [6].

2.4 Présentation des différentes tâches

Il existe de nombreuses tâches de traitement du langage naturel (NLP) que l'on peut appliquer à diverses langues, notamment l'arabe. Ces tâches incluent :

- **Reconnaissance d'Entités Nommées (NER)**

La reconnaissance des entités nommées est une tâche qui consiste à identifier et à classer des entités spécifiques telles que les personnes, les organisations ou les lieux dans un texte. Elle a également été appliquée au dialecte algérien dans plusieurs travaux (par exemple [7] et [8])

- **Réponse aux questions (QA)**

Il s'agit de concevoir des systèmes capables de comprendre une question en langage naturel et d'y répondre de manière précise à partir d'un texte source. Parmi les systèmes conversationnels

intégrant des capacités de répondre aux questions, on peut citer DZChatbot [9], qui a été utilisé avec succès pour répondre aux questions des utilisateurs dans le domaine médical en dialecte algérien.

- **Traduction automatique**

La traduction automatique est le processus qui permet de convertir un texte d'une langue à une autre sans intervention humaine. Par exemple, l'étude [10] propose le fine-tuning de AraT5 pour traduire des textes multidialectaux, y compris le dialecte algérien, vers l'arabe standard moderne.

- **La modélisation thématique (Topic Modeling)**

La modélisation thématique vise à identifier automatiquement les thèmes ou les catégories dominants (par exemple, le sport, la politique, l'économie) abordés dans un texte. Facilitant ainsi l'organisation et la recherche d'information. Par exemple, dans [11], cette approche a été utilisée pour catégoriser des commentaires en dialecte algérien publiés sur les réseaux sociaux.

- **Analyse de sentiment**

L'analyse de sentiment vise à détecter l'attitude exprimée dans un texte (positive, négative, neutre). C'est l'une des tâches les plus étudiées dans le cadre du dialecte algérien. Diverses approches ont été développées, pour réaliser cette tâche, allant des systèmes basés sur des règles jusqu'aux modèles basés sur l'apprentissage approfondi (deep learning) comme les Large Langage Model (LLM).

3 Méthodes utilisées en analyse de sentiment

Comme mentionné précédemment parmi les tâches courantes du traitement du langage naturel (NLP), l'analyse de sentiments occupe une place centrale.

Il existe différentes méthodes permettant de comprendre et de classifier les sentiments exprimés dans les textes, que nous présentons ci-dessous : les approches basées sur des lexiques, les approches d'apprentissage automatique (Machine Learning), les approches d'apprentissage profond, ainsi que les modèles de langage (LLM).

3.1 Les approches basées sur des lexiques

Les approches basées sur les lexiques reposent sur des ressources linguistiques prédéfinies pour analyser la polarité d'un texte. Elles s'appuient sur deux types principaux d'approches :

- **Approche basée sur les corpus :** Cette méthode exploite de grands corpus textuels annotés pour détecter les sentiments en se basant sur des motifs syntaxiques et des similarités contextuelles. Elle commence par une liste initiale, puis identifie et étend les termes sentimentaux selon leur orientation. Son efficacité dépend fortement de la taille et de la qualité du corpus.
- **Approche basée sur les dictionnaires :** Elle consiste à utiliser un lexique prédéfini de mots associés à des scores de sentiment (positif, négatif ou neutre). Le score global d'un texte est obtenu en agrégeant les scores des mots qu'il contient. Cette approche simple à mettre en œuvre peut être enrichie par des synonymes et antonymes issus de ressources comme WordNet [12].

3.2 Machine Learning (ML)

L'émergence du machine learning a marqué une avancée significative par rapport aux approches mentionnées précédemment, en permettant aux modèles d'apprendre directement à partir des données. En analyse des sentiments, les approches de machine learning sont généralement classées en trois catégories principales :

- **Apprentissage supervisé :** Il reste le plus utilisé. Il repose sur des modèles entraînés à partir de données annotées pour reconnaître des motifs et prédire le sentiment d'un texte. Parmi les algorithmes les plus courants, on trouve le Naive Bayes, Machine à Vecteurs de Support (SVM), régression logistique (LR), arbres de décision (DT), k plus proches voisins (k-NN) et forêts aléatoires, souvent associés à des représentations comme Bag-of-Words ou TF-IDF [13]**.**
- **Apprentissage non supervisé :** Cette méthode ne nécessite pas de données annotées. Elle repose sur des techniques comme le clustering, qui permettent de regrouper les textes selon leur similarité. Toutefois, elle présente des limites, notamment le besoin d'un volume important de données pour s'entraîner de manière précise**.**
- **Apprentissage semi-supervisé :** Il combine des données annotées et non annotées, ce qui le rend utile lorsque l'annotation manuelle est limitée. Il utilise des algorithmes comme le self-training, le co-training ou les méthodes basées sur les graphes [14].

3.3 Deep Learning (DL)

Les modèles d'apprentissage profond (ou Deep Learning en anglais) reposent sur l'approche des réseaux de neurones artificiels, inspirée du fonctionnement des réseaux neuronaux

biologiques observés chez l'être humain. Ces modèles apprennent à partir des données fournies d'une manière similaire à celle du cerveau humain, Ces modèles apprennent à partir des données fournies d'une manière similaire à celle du cerveau humain, en ajustant les poids et en s'entraînant de façon itérative sur plusieurs époques [15].

Parmi les architectures les plus répandues, on trouve les réseaux de neurones récurrents (RNN), les réseaux de mémoire à long et court terme (LSTM), qui sont une variante des RNN, ainsi que les réseaux de neurones convolutifs (CNN) [16].

Plus récemment, les modèles basés sur l'architecture Transformer comme BERT (Bidirectional Encoder Representations from Transformers) et GPT (Generative Pre-trained Transformer) ont marqué une avancée majeure grâce à leur capacité à comprendre le contexte global d'un texte et à le générer avec une grande précision, en particulier lorsqu'ils sont adaptés à des corpus spécifiques.

3.4 Large Langage Model (LLM)

3.4.1 Définition

Les LLMs sont des systèmes de traitement du langage naturel (NLP) entraînés sur de vastes ensembles de données capables de comprendre et de générer du texte, Ils sont conçus pour accomplir diverses tâches linguistiques, telles que la traduction, la réponse à des questions et l'analyse des sentiments. Grâce à leurs capacités avancées d'apprentissage des représentations complexes de la langue.

Les modèles de langage de grande taille (LLMs, pour Large Language Models) utilisent une architecture appelée Transformer (voir Figure 1.1). Ils sont entraînés sur d'immenses quantités de textes, ce qui leur permet de comprendre le langage humain et de produire des textes cohérents et adaptés au contexte.

L'élément central de cette architecture est le mécanisme d'attention, plus précisément la self-attention. Ce mécanisme a été introduit dans l'article Attention is All You Need [17]. Il permet au modèle de repérer, à chaque étape, les mots les plus importants dans l'ensemble d'une phrase ou d'un texte.

Grâce à cette capacité, les LLM peuvent réaliser de nombreuses tâches en lien avec la langue : traduction automatique, analyse de sentiments, réponse à des questions, génération de texte, ou encore classification de documents [2].

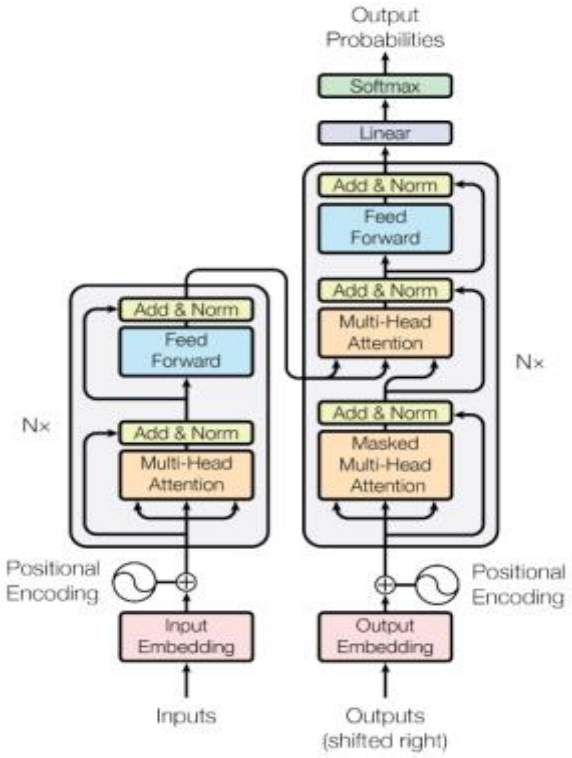

Figure 1. 1. Architecture transformer [17]

3.4.2 Architectures des LLMs

Les architectures à la base des LLMs se répartissent en trois grandes catégories : encodeur, décodeur et encodeur-décodeur. Chacune remplit un rôle spécifique dans les tâches de traitement du langage, influençant la manière dont les modèles comprennent et génèrent du texte [2].

- **Décodeurs Uniquement**

Les modèles décodeurs uniquement (Decoder-Only) sont conçus pour générer du texte et traduire automatiquement, en produisant une sortie cohérente à partir de représentations internes. Ces modèles sont largement utilisés en modélisation du langage (Language Modeling), traduction et génération de dialogue. À titre d'exemple, les modèles de type GPT, tels que GPT-2, GPT-3 et GPT-4, ainsi que des modèles similaires comme LLaMA (Meta) et Gemma (Google), utilisent un mécanisme unidirectionnel pour prédire chaque mot en fonction du contexte précédent. GPT-2 (1,5 milliard de paramètres) a marqué une avancée en génération de

texte, suivi par GPT-3 (175 milliards de paramètres), qui améliore les performances en réponse aux questions et en traduction.

Des versions arabes, comme AraGPT2 et JASMINE, ont également été développées.

- **Encodeurs Uniquement**

Les modèles encodeurs uniquement (Encoder-Only) sont conçus pour des tâches nécessitant une compréhension approfondie du texte, telles que la classification et la prédiction de phrases. BERT est l'un des modèles les plus influents de cette catégorie. Il fonctionne de manière bidirectionnelle, lui permettant d'apprendre le contexte des mots en analysant simultanément ceux qui le précèdent et le suivent. Ils excellent notamment dans des tâches comme la classification de texte.

Le succès de BERT a conduit au développement de plusieurs modèles arabes tels que AraBERT et MARBERT, ainsi que de modèles dialectaux comme **DarijBERT**, dédié au dialecte marocain.

- **Encodeurs-Décodeurs**

Les modèles encodeurs-décodeurs, comme le Text-to-Text Transfer Transformer (**T5**), sont conçus pour traiter des tâches nécessitant à la fois la compréhension et la génération de texte. Ils adoptent un format textuel unifié pour toutes les tâches NLP et améliorent l'apprentissage en masquant des séquences entières de mots. Des modèles arabes comme **AraMU**S et **AraT5** exploitent cette architecture pour améliorer le traitement du texte en arabe

3.4.3 Fine-tuning des LLMs

Le fine-tuning consiste à utiliser un modèle pré-entraîné comme base, puis à poursuivre son entraînement sur un nouveau jeu de données ciblé pour une tâche spécifique, cette approche exploite les connaissances déjà acquises par le modèle pour les adapter cette tâche.

Le fine-tuning des LLMs aux dialectes constitue un défi majeur en (NLP). Contrairement aux langues standardisées, les dialectes se caractérisent par un manque de ressources, une grande variabilité linguistique et une absence de normalisation.

Le fine-tuning des modèles de langage de grande taille (LLMs) peut être appliqué aussi bien aux données en arabe dialectal qu'à celles en arabe standard. Parmi les modèles ayant été adaptés à des tâches spécifiques, on retrouve AraBERT (basé sur une architecture à encodeur), utilisé pour l'analyse de sentiments [2].

On peut également citer AraGPT, un modèle à décodeur uniquement, conçu pour la tâche de question answering, ainsi que AraT5, un modèle encodeur-décodeur, employé pour plusieurs tâches, notamment la traduction automatique.

4 Travaux connexes

Depuis quelques années, plusieurs travaux de recherche ont été menés autour du traitement du dialecte arabe, et en particulier du dialecte algérien, dans diverses tâches du NLP. Ces études se répartissent selon les approches traditionnelles, les méthodes de deep learning, et l'utilisation récente des LLMs. Cette section présente une synthèse des contributions majeures dans le domaine, classant selon la tâche abordée, en met l'accent sur la tâche d'analyse de sentiments et quelque autre tâche tel que détection d'émotions, classification de sujets, traduction automatique et reconnaissance des entités nommées.

4.1 Analyse de sentiment

4.1.1 Méthode lexicale

Mataoui et al.[18] ont proposé une approche basée sur des lexiques pour l'analyse de sentiments en arabe algérien. Trois lexiques ont été construits : un lexique de mots-clés, un lexique de négations et un lexique de mots d'intensification, tous annotés avec des polarités de sentiment. En complément, une liste d'émoticônes et un dictionnaire de phrases courantes ont également été intégrés. Leur approche lexicale, testée sous différentes configurations, a atteint une accuracy de 79,13 %.

4.1.2 Machine learning

Cet article [19] propose une approche automatique pour la construction d'un corpus annoté destiné à l'analyse des sentiments avec le dialecte algérien. Le corpus contient 8 000 messages, répartis équitablement entre l'arabe et l'Arabizi, et a été évalué à l'aide de plusieurs modèles de classification (SVM, NB, LR, DT, RF) pour chaque méthode de vectorisation (BOW, Doc2vec). Les résultats expérimentaux ont montré que LR avec BOW a donné les meilleurs résultats, avec un F1 score atteignant 72 % pour les messages en arabe et 78 % pour ceux en Arabizi.

Abdelli et al.[20] ont construit un corpus d'analyse de sentiment en arabe algérien et en arabe standard moderne, composé de 49 864 commentaires annotés de manière équilibrée : 24 932

positifs et 24 932 négatifs. Après un nettoyage, incluant la suppression du contenu non arabe et la normalisation des caractères arabes, ce corpus a été utilisé pour entraîner un modèle SVM basé sur la représentation TF-IDF. Les résultats obtenus montrent que le SVM atteint une accuracy de 86 %, démontrant une performance supérieure à d'autres approches comme le LSTM.

Dans cette étude [21], les auteurs ont construit un jeu de données annoté de 11 760 commentaires en dialecte algérien, collectés à partir de différentes plateformes de réseaux sociaux. Plusieurs algorithmes de Machine Learning ont été appliqués, notamment Naïve Bayes (Bernoulli, Multinomial, Gaussien) et SVM. Le meilleur résultat a été obtenu par MNB avec une accuracy de 84,21 % sur les données transcrites, suivi de SVM avec 83,70 % sur les données enrichies par Word2vec.

4.1.3 Deep learning :

Dans cette étude [22], un corpus de 25 475 commentaires annotés manuellement a été utilisé. Après une phase de prétraitement du corpus, deux modèles de réseaux de neurones, à savoir un perceptron multicouche (MLP) et un réseau de neurones convolutif (CNN), ont été entraînés pour classer les commentaires selon trois catégories : négatif, neutre ou positif. Les auteurs ont obtenu une accuracy de 89,5 % avec le modèle CNN et de 81,6 % avec le modèle MLP.

L'étude de Mazari et al. [23] montre l'efficacité positive des modèles CNN et RNN pour classer des textes d'opinions rédigés en dialecte algérien, extraits des réseaux sociaux à propos du Hirak_19. L'expérimentation a été réalisée sur un corpus de 7800 commentaires. Ils ont obtenu une accuracy améliorée pour la classification des sentiments, avec 63,28% pour le modèle CNN et 60,97% pour le modèle RNN.

Moudjari et al. [24] ont présenté "TWIFIL", une plateforme ouverte dédiée à l'annotation publique des commentaires Twitter. Cette plateforme a permis de constituer un corpus annoté de 9 000 tweets en dialecte algérien. Pour évaluer la qualité de ce corpus, plusieurs modèles de deep learning ont été testés, parmi lesquels les modèles CNN et LSTM ont obtenu les meilleures performances, avec accuracy respectif de 76 % et 75 %.

4.1.4 Large Language Model (LLM)

DziriBERT [5] est un modèle de langage pré-entraîné spécifiquement pour l'arabe algérien, basé sur l'architecture BERT et conçu pour surmonter les limites des modèles arabes et

multilingues existants. À cet effet, les auteurs utilisent un corpus de 1,1 million de tweets algériens**.** Lors de son évaluation dans le cadre de l'analyse des sentiments, en utilisant les corpus publics TWIFIL et NARABIZI**.** DziriBERT a surpassé les modèles précédemment existants. En particulier pour l'Arabizi, il a atteint un F1 score de 80,5 % avec TWIFIL et de 61,2 % avec NARABIZI, dépassant ainsi MARBERT**.** Bien que son corpus d'entraînement soit relativement limité. DziriBERT démontre que les modèles spécialisés peuvent offrir d'excellentes performances, même avec des ressources réduites.

L'étude [25] présente AlgBERT, un modèle basé sur AraBERT pour l'annotation automatique de corpus en dialecte algérien, destiné à l'analyse des sentiments. Elle s'appuie sur trois sources principales de données : le corpus DZSentiA, une partie extraite de DziriBERT, ainsi qu'environ 7 000 commentaires collectés manuellement, rédigés en arabe standard (MSA) et en dialecte algérien. L'ensemble a permis de constituer un corpus de 54 000 commentaires annotés manuellement en deux classes : positif et négatif. Les résultats obtenus affichent une accuracy de 92,6 %**,** grâce notamment à l'intégration d'émojis dans l'analyse. Cette solution marque un progrès important pour le traitement du dialecte algérien.

L'étude de Benmounah et al.[26] repose sur la création d'un jeu de données de 45 000 commentaires extraits de chaînes YouTube algériennes, annotés manuellement en trois classes : négatif, neutre et positif. Après le prétraitement, différents modèles ont été ajustés (fine-tuned), notamment des variantes du modèle BERT ainsi que des modèles de deep learning classiques comme LSTM et BiLSTM. La version BERT Arabic Large a montré les meilleures performances atteignant un F1 score de 78,38 % et une accuracy de 81,7 %. Ces résultats confirment l'efficacité des Transformers pour l'analyse des sentiments en dialecte algérien.

L'article [27] présente FASSILA un corpus spécialisé développé pour la détection de fake news et l'analyse de sentiment dans le dialecte algérien. La méthodologie implique la collecte de données issues de Facebook, YouTube et des corpus existants, suivie d'un nettoyage et d'une augmentation des données. Le corpus final comprend 10 087 phrases. Plusieurs modèles de classification ont été évalués, incluant des Transformers pré-entraînés (AraBERT, MARBERT, DziriBERT). Les résultats montrent que AraBERT excellent en analyse des sentiments avec un accuracy de 83,92 %. Le corpus est disponible gratuitement, facilitant son utilisation par d'autres chercheurs dans ce domaine.

Le tableau 1.2 présente un aperçu complet des recherches antérieures menées dans différentes méthodes utilisées dans l'analyse des sentiments dans dialecte algérien.

Etude	Années	Data	Approche	Meilleur résultat
[18]	2016	Commentaire et post de facebook	Lexique	Accuracy 79,13 %.
[19]	2018	8 000 messages en dialecte algérien	SVM, NB, LR, DT, RF	F1 score 78% avec LR
[22]	2018	25 475 commentaires	MLP, CNN	Accuracy 89,5% avec CNN
[20]	2019	49 864 commentaires (Algérien + MSA)	SVM, LSTM	Accuracy 86 % avec SVM
[24]	2020	9 000 tweets en dialecte algérien	LSTM, CNN	Accuracy 87% avec CNN
[5]	2020	1,1 million de tweets algériens	Pré-entraînement (DziriBERT)	F1 score de 80,5 % avec TWIFIL
[23]	2021	7800 commentaires Algérien	CNN, RNN	Accuracy 60,97% avec RNN
[21]	2022	11 760 commentaires	BNB, MNB, GNB et SVM.	Accuracy 84,21 % avec MNB
[25]	2023	54000 commentaire algérien	Fine-tuning (AraBERT)	Accuracy 92,6 %
[26]	2023	45 000 commentaires YouTube algérien	Fine-tuning différentes variantes du modèle BERT en arabe	Accuracy : 81.7% avec BERT Arabic Large
[27]	2024	10 087 phrases algérien (Facebook, YouTube et corpus existants)	Fine-tuning (AraBERT, MARBERT, DziriBERT)	Accuracy 83,92 % avec AraBERT

Tableau 1. 2. Travaux d'analyse de sentiments appliqués au dialecte algérien

4.2 Autres tâches dans NLP

4.2.1 Classification des émotions

Le travail de Rehaiem et Dida [28] vise à détecter les émotions dans des textes issus des réseaux sociaux en dialecte algérien, en utilisant des techniques de clustering non supervisées. Les données ont été collectées via l'API Twitter, puis soumises à un prétraitement complet (nettoyage, tokenisation, vectorisation par TF-IDF). Le traitement principal repose sur l'algorithme NMF (Non-negative Matrix Factorization) pour regrouper les textes. Les émotions extraites suivent le modèle d'Ekman, réparties en six catégories : bonheur, colère, peur, surprise, tristesse et dégoût. Le système a également bien géré les emojis, traduits en mots-clés émotionnels. Ce travail a abouti à la création d'un corpus annoté de qualité, pouvant servir de base à l'entraînement de modèles supervisés.

4.2.2 Traduction automatique

L'article [29] propose une approche de transductive transfer learning pour améliorer la traduction neuronale du dialecte algérien à faible ressource. Deux modèles NMT ont été affinés (fine-tuned) Seq2Seq et Attentional Seq2Seq, initialement entraînés sur le corpus MADAR, puis adaptés sur PADIC. Les résultats montrent une nette amélioration grâce au transfert : le score BLEU passe de 0.3 à 34.56 pour Seq2Seq, et de 16.5 à 35.87 pour Attentional Seq2Seq. Ces performances confirment l'efficacité du transfert transductif pour la traduction du dialecte algérien.

4.2.3 Classification de sujet

Dans le cadre de la classification de sujets, le travail [30] présente une approche de détection des sujets dans les publications en dialecte algérien à des fins d'analyse marketing. Un corpus de 1 000 postes depuis Facebook et Jumia a été annoté selon 11 catégories (accesoires, bijoux, électroménager, alimentation, etc.). La méthode combine l'extraction des mots-clés propres à chaque classe avec TF-IDF et l'utilisation d'un modèle de deep learning pour la classification. Le système a atteint une précision de 83,03 %, démontrant ainsi son efficacité pour la classification des publications en dialecte algérien.

Il y'a aussi le travail de Touileb et Barnes [11] qui présente un corpus multi-script annoté en dialecte algérien : NArabizi, arabe algérien (DZ) et code-switché, pour la classification de

sujets. Plusieurs modèles ont été évalués, notamment BOW, CNN et BiLSTM. Le modèle BiLSTM obtient les meilleurs résultats sur DZ avec un score F1 de 57 %. Ces résultats montrent la difficulté du transfert entre scripts pour la classification thématique.

4.2.4 Reconnaissance des entités nommées

En 2023 Dahou et Cheragui présentent DzNER [31] présentent DzNER**,** un ensemble de données algérien dédié à la reconnaissance des entités nommées, composé de plus de 21 000 phrases, extraits de Facebook et de chaînes YouTube algériennes. Le processus d'annotation manuelle a été réalisé par deux annotateurs spécialisés sur le dialecte algérien, visant à identifier trois types d'entités : Personnes (PER), Organisations (ORG) et Lieux (LOC). Pour évaluer l'efficacité du corpus, les auteurs ont entraîné les modèles AraBERT et DziriBERT sur DzNER, puis les ont testés sur le dataset NArabizi. Les résultats montrent que AraBERT obtient un score F1 de 75,41 % tandis que DziriBERT atteint 74,69 %, confirmant la pertinence de DzNER comme ressource essentielle pour le traitement du dialecte algérien en NLP, notamment pour la tâche de NER.

Un travail similaire [8] explore l'efficacité des modèles pré-entraînés. La méthodologie consiste à fine-tuning plusieurs modèles pré-entrainés (AraBERT, MARBERT, mBERT et QARiB) en les entraînant sur le corpus ANERCorp, puis en les testant sur ANERCorp et DzNER. AraBERT obtient les meilleurs résultats avec un score F1 de 85,57 % sur ANERCorp et de 85,52 % sur DzNER, confirmant son efficacité pour la tâche de reconnaissance des entités nommées en dialecte algérien.

Le tableau suivant 1.3 présente un aperçu des recherches antérieures menées dans différentes tâches et méthodes dans dialecte Algérien.

Etude	Années	Data	Tache	Approche	Meilleur résultat
[28]	2021	3000 Commentaires tweeter	Détection des émotions	Clustering non supervisé (NMF),	Pas précisé
[29]	2021	MADAR (52 000 phrases), PADIC (12 800 phrases du dialecte algérien)	Traduction automatique	Transductive Transfer Learning (NMT)	BLEU 35.87
[11]	2021	Corpus multi-script en dialecte Algérien	Classification de sujet	BOW, CNN, BiLSTM.	F1 score 57% avec BiLSTM
[31]	2023	21 000 phrases, de Facebook et YouTube	Reconnaissance d'entité nommée	Fine tuning (AraBERT, DziriBERT)	F1 score 75,41 % avec AraBERT
[8]	2023	Corpus DzNER		Fine tuning (AraBERT, MARBERT, mBERT, QARiB)	F1 score 85,57 % avec AraBERT
[30]	2024	1 000 publications Facebook et Jumia	Classification de sujet	Représentation vectorielle TF-IDF, deep learning	Précision 83 %

Tableau 1. 3.Etudes avec différentes tâches et méthode en dialecte Algérien

5 Conclusion

Dans ce chapitre, nous avons abordé les concepts fondamentaux, en commençant par les différentes variantes de la langue arabe, en particulier le dialecte algérien. Nous avons également présenté un aperçu des méthodes utilisées en analyse de sentiment, allant des approches lexicales aux modèles de type LLM. Enfin, nous abordé quelques synthèses des travaux précédents portant sur le dialecte algérien.

Dans le prochain chapitre, nous présenterons en détail notre méthodologie de travail.

Chapitre 2 : Méthodologie

1 Introduction

Dans ce chapitre, nous allons présenter l'approche proposée ainsi que les différentes étapes nécessaires à la mise en place d'un système d'analyse de sentiments appliqué aux commentaires rédigés en dialecte algérien. Nous détaillons ci-dessous chaque étape de notre méthodologie afin de mettre en évidence son impact sur la performance globale du système.

2 Architecture générale

Notre contribution est présentée dans la Figure 2.1, qui illustre l'architecture générale adoptée dans notre méthodologie pour la classification des sentiments en dialecte algérien.

Le processus commence par la collecte des données, une étape essentielle durant laquelle des jeux de données existants ont été sélectionnés et consolidés pour former une base cohérente. Cette phase a été suivie d'un prétraitement approfondi, visant à nettoyer les textes en supprimant les éléments non pertinents afin de réduire le bruit et d'améliorer la qualité des données.

Une fois le texte nettoyé, il est soumis à une tokenisation, c'est-à-dire découpé en unités linguistiques adaptées aux modèles de langage. Cette opération, spécifique à chaque modèle pré-entraîné, permet de transformer le texte en une représentation numérique exploitable.

Vient ensuite l'étape de fine-tuning, au cours de laquelle chaque modèle est ajusté aux spécificités du dialecte algérien à l'aide de notre jeu de données.

Enfin, une phase d'évaluation est réalisée afin de mesurer les performances des modèles, suivie de l'exploitation du meilleur modèle dans un système de classification de sentiments intégré à une application réelle.

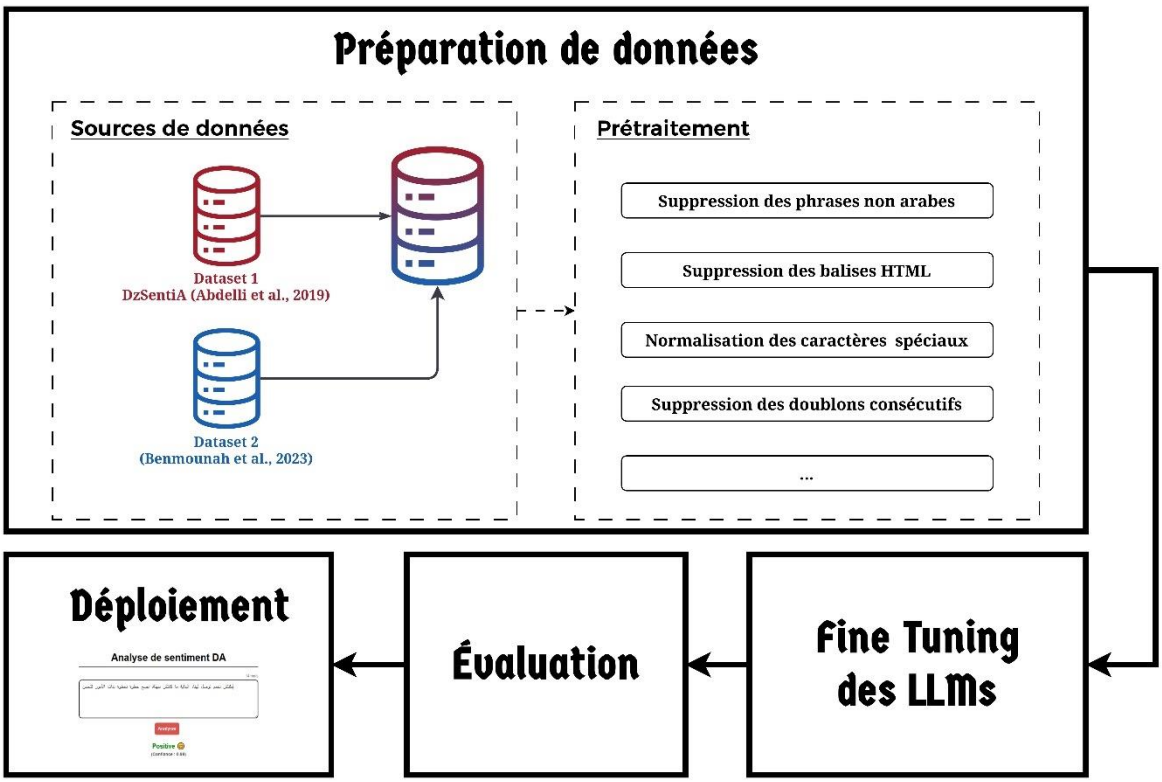

Figure 2. 1. Architecture générale.

3 Méthodologie

3.1 Sources de données

Dans le cadre de notre approche, nous avons intégré le contenu de deux ensembles de données distincts, issus de sources différentes, afin de constituer un corpus homogène adapté à notre tâche de classification des sentiments :

Le premier jeu de données utilisé est un corpus constitué de 45000 commentaires extraits de chaînes YouTube algériennes et annotés manuellement en trois classes (positif, neutre ou négatif). Ce corpus a été recueilli et préparé par Zakaria Benmounah [26].

Le deuxième jeu de données utilisé est DzSentiA, un corpus composé de 49864 commentaires, répartis équitablement entre 24932 commentaires positifs et 24932 commentaires négatifs. Ces données ont été collectées à partir de pages Facebook algériennes populaires [20].

3.2 Prétraitement

Le prétraitement des données est une étape essentielle dans toute tâche de NLP y compris la classification des sentiments. Il permet de nettoyer les données brutes en supprimant les éléments inutiles et en standardisant les textes, ce qui facilite l'apprentissage et la généralisation des modèles, même les plus avancés comme les LLMs, qui restent sensibles à la qualité des données d'entrée.

Cette étape est d'autant plus cruciale pour les langues riches comme l'arabe, ou pour ses variantes dialectales (dialecte algérien), où les normes d'écriture sont souvent instables, avec des orthographes et formes linguistiques très variées. Un prétraitement adapté contribue à réduire cette variabilité, à uniformiser les formes linguistiques et à améliorer la cohérence des textes analysés.

Pour cela et après une analyse approfondie de nos données, nous avons ainsi appliqué les étapes suivantes :

- **Filtrage des textes par nombre de mots :** Les phrases contenant moins de 4 mots ou plus de 128 mots ont été supprimées, car elles sont soit trop brèves pour être informatives, soit trop longues et peu nombreuses.
- **Suppression des phrases non arabes :** Notre ensemble de données est filtré à partir des commentaires écrits en caractères non arabes pour assurer la cohérence linguistique et améliorer les performances des modèles.
- **Suppression des balises HTML :** Certains textes contiennent des balises HTML (par exemple,
). Ces éléments ont été supprimés pour obtenir un texte brut et propre.
- **Suppression de tous types d'emojis** : Les emojis ont été éliminés, car dans le cadre de la classification des sentiments basée uniquement sur le texte, ils ne sont pas considérés comme essentiels à l'analyse.
- **Normalisation des caractères arabes spéciaux :** Certains caractères arabes ont été remplacés par d'autres plus courants, comme la conversion de آ, أ et إ en ا, afin de réduire la variabilité orthographique et faciliter la généralisation des modèles.
- **Suppression des chiffres :** Tous les chiffres ont été supprimés, car ils n'apportent pas d'information sémantique utile dans le contexte de l'analyse de sentiment.
- **Suppression de la ponctuation :** La ponctuation a été retirée pour uniformiser les textes et éviter les tokens non pertinents.

- **Suppression des doublons consécutifs :** Les mots répétés de manière consécutive, ont été réduits à une seule occurrence pour éviter la redondance émotionnelle ou les répétitions inutiles.
- **Normalisation des espaces :** Les espaces en trop ont été supprimés pour assurer une séparation cohérente entre les mots, ce qui facilite la tokenisation ultérieure.

À la fin de cette phase de prétraitement, le jeu de données a été réparti en trois sous-ensembles (70 % pour l'entraînement**,** 15 % pour la validation, 15 % pour le test)

3.3 Tokenisation

La tokenisation est une étape clé du traitement automatique du langage naturel. Elle consiste à diviser le texte en unités plus petites appelées tokens, qui peuvent correspondre à des mots, des sous-mots ou des caractères. Cette étape permet de transformer le texte brut en une représentation compréhensible par les modèles de langage.

Dans notre travail, la tokenisation a été effectuée à l'aide des tokenizers pré-entraînés fournis avec chaque modèle LLM utilisé, garantissant une cohérence optimale entre le prétraitement et le modèle.

3.4 Représentation vectorielle (embeddings)

Après la tokenisation, les textes sont convertis en vecteurs numériques appelés embeddings, qui peuvent être traités par une machine. Ces représentations capturent la signification sémantique des mots, sous-mots ou phrases, en tenant compte de leur contexte d'apparition.

Dans notre méthodologie, nous utilisons des embeddings contextuels générés automatiquement par les modèles de langage pré-entraînés (LLM) sélectionnés. Ces représentations, apprises durant la phase de pré-entraînement, sont ensuite ajustées (fine-tuned) sur notre tâche spécifique de classification des sentiments.

Ces vecteurs constituent l'entrée directe des phases de fine-tuning et de classification. La qualité de cette représentation est déterminante, car elle conditionne la capacité du modèle à généraliser, à distinguer les différentes polarités de sentiment, et à interpréter correctement des formulations complexes ou dialectales.

3.5 Fine-tuning des modèles

Le fine-tuning des modèles pré-entraînés a été réalisé sur notre corpus d'entraînement en dialecte algérien. Cette étape vise à adapter les représentations linguistiques générales, apprises

lors du pré-entraînement, à la tâche spécifique de classification binaire des sentiments (positif et négatif).

- **Modèles spécialisés pour l'arabe standard et dialectal :**
  - **AraBERT [32] :** AraBERT est le premier modèle BERT spécifiquement conçu pour l'arabe standard moderne (MSA). Il a été pré-entraîné sur un corpus d'environ 70 millions de phrases (~24 Go de données), avec 110 millions de paramètres et un vocabulaire de 64 000 tokens. Il a démontré de fortes performances sur plusieurs tâches NLP en arabe standard.

  - **QARiB (QCRI Arabic and Dialectal BERT) [33] :** est un modèle BERT pré-entraîné par QCRI (Qatar Computing Research Institute), conçu pour le traitement de l'arabe standard et des dialectes arabes. Il a été entraîné sur un corpus de 180 millions de phrases, représentant environ 14 milliards de tokens, avec un vocabulaire de 64 000 tokens. Le modèle repose sur l'architecture BERT-base et bénéficie d'un entraînement sur des données couvrant de nombreuses variétés dialectales. Grâce à la richesse et la diversité de son corpus, QARiB obtient de bonnes performances sur les tâches NLP en arabe dialectal

  - **CAMeLBERT-DA [34]** est une version dialectale de CAMeLBERT, pré-entraînée sur un large corpus de textes arabes dialectaux issus de diverses régions. Il repose sur l'architecture **BERT-base** avec **110 millions de paramètres**, un vocabulaire de **30 000 tokens**, et une taille sur disque de **439 MB**. Le modèle est conçu pour capturer les spécificités linguistiques des dialectes arabes. Il est particulièrement adapté à la classification de sentiments en contexte dialectal, notamment pour l'arabe algérien.

- **Modèles multilingues** :

  - **XLM-R (XLM-RoBERTa) [35] :** Est un modèle multilingue basé sur l'architecture RoBERTa, Pré-entraîné sur 2,5 To de texte couvrant 100 langues. Il utilise un vocabulaire partagé de 250 000 tokens et compte environ 270 millions de paramètres dans sa version base. Grâce à son entraînement sur des données massif et diversifié, XLM-R excelle dans les tâches de NLP

multilingue, y compris celles portant sur des langues à ressources limitées comme l'arabe dialectal .

- **mBERT (Multilingual BERT) [36] :** mBERT est le modèle BERT multilingue entraîné sur 104 langues, y compris l'arabe. Il repose sur l'architecture BERT-base avec environ 179 millions de paramètres, un vocabulaire de 119 547 tokens, et une taille sur disque de 714 Mo. Bien qu'il ne soit pas spécialisé pour l'arabe, il montre de bonnes performances sur les dialectes grâce à sa large couverture linguistique.
- **DistilBERT [37] :** DistilBERT est une version compressée de mBERT. Il contient 135 millions de paramètres, utilise le même vocabulaire de 119 547 tokens, et sa taille sur disque est réduite à 543 Mo. Conçu pour être plus rapide et plus léger que mBERT, il conserve des performances solides sur les tâches de NLP, comme la classification de sentiments.

Afin d'obtenir les meilleurs résultats possibles pour chaque modèle, nous avons ajusté différents hyperparamètres, notamment :

- Le taux d'apprentissage (learning rate).
- La taille des batchs (batch size).
- Le nombre d'époques (epochs).
- L'utilisation de la stratégie d'arrêt anticipé (early stopping) pour éviter le surapprentissage.

3.6 Évaluation

Après l'entraînement des modèles, nous avons procédé à l'évaluation de leurs performances en utilisant plusieurs métriques standards sur l'ensemble de test :

- **Accuracy :** L'accuracy est la métrique la plus couramment utilisée pour les tâches de classification. Elle mesure la proportion de prédictions correctes parmi l'ensemble des prédictions effectuées par le modèle [12].

$$Accuracy = \frac{TP + TN}{T\,P \;+\; FP \;+\; FN \;+\; TN}$$

- **Précision :** La précision est une mesure qui évalue la proportion de pré dictions positives correctes parmi toutes les prédictions positives effectuées par le modèle.

$$Précision = \frac{TP}{TP + FP}$$

- **Rappel (Recall) :** Cette métrique est complémentaire à la précision, car elle permet d'évaluer la qualité des prédictions positives qui sont effectivement correctes [16]**.**

$$Recall = \frac{T\,P}{TP + FN}$$

- **F1-score :** Le F1 score est la moyenne harmonique de la précision et du rappel. Il est particulièrement utile lorsque les classes sont déséquilibrées, car il prend en compte à la fois les faux positifs et les faux négatifs [12]**.**

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Afin d'évaluer la capacité de généralisation des modèles, nous les avons testés sur d'autres jeux de données provenant de dialectes différents, dans le but d'évaluer leur capacité à s'adapter à des variantes dialectales non vues lors de l'entraînement. Cette étape permet de tester la robustesse des modèles face à la diversité linguistique de l'arabe dialectal, et de vérifier s'ils peuvent généraliser efficacement au-delà du dialecte algérien :

- **Dialecte marocain :** nous avons utilisé le corpus MSAC (Moroccan Sentiment Analysis Corpus), créé par Oussous et al. [38], composé de 2 000 textes marocains écrits en caractères arabes, annotés en deux polarités : positive et négative.
- **Dialecte tunisien :** nous avons utilisé le corpus TSAC (Tunisian Sentiment Analysis Corpus), contenant 17 000 commentaires annotés manuellement en polarités positive et négative. Ce corpus a été collecté à partir de commentaires d'utilisateurs Facebook sur les pages officielles de radios et chaînes de télévision tunisiennes [39].

  .

Avant d'être utilisés dans la phase d'évaluation, ces deux jeux de données ont été nettoyés selon les mêmes étapes de prétraitement que celles appliquées à notre corpus principal, et la tokenisation a été réalisée à l'aide des mêmes modèles de langage pré-entraînés (LLM)**.** Cela a permis d'unifier le format des données et de garantir la cohérence des entrées pour l'évaluation des modèles.

3.7 Déploiement

Après l'évaluation de nos modèles, nous avons sélectionné celui offrant les meilleures performances pour une utilisation en production. Ce modèle a ensuite été intégré dans un site web, permettant aux utilisateurs de saisir un texte en dialecte algérien et d'obtenir une analyse de sentiment.

4 Conclusion

Dans ce chapitre, nous décrivons en détail notre méthodologie de fine-tuning de différents modèles de langage (LLM) pour la tâche d'analyse de sentiments en dialecte algérien. Cette explication couvre l'ensemble des étapes, depuis la préparation des données jusqu'à l'évaluation à l'aide de jeux de données variés.

Dans le chapitre suivant, nous aborderons l'application pratique de cette méthodologie et discuterons des résultats obtenus

Chapitre 3 : Implémentation et résultats

1 Introduction

En s'appuyant sur la méthodologie présentée dans le chapitre précédent, ce chapitre se concentre sur la mise en œuvre pratique de notre recherche. Nous commençons par décrire l'environnement de travail, en précisant le langage de programmation utilisé ainsi que les différentes bibliothèques essentielles à notre approche. Des extraits de code illustratifs sont ensuite proposés afin de mettre en lumière les éléments clés de l'implémentation.

Par la suite, les résultats obtenus sont présentés, accompagnés d'une analyse des expérimentations menées. Cette exposition détaillée vise à offrir une compréhension claire et précise de l'exécution pratique et des résultats empiriques de notre étude.

2 Environnement de travail (Kaggle)

Kaggle est une plateforme en ligne de science des données, appartenant à Google, qui permet aux chercheurs, étudiants et praticiens de collaborer, de partager des jeux de données et de participer à des compétitions de machine learning. Elle offre un environnement basé sur des notebooks Jupyter exécutables dans le cloud, avec un accès gratuit à des GPU (jusqu'à 30 heures par semaine) et TPU (jusqu'à 20 heures par semaine). Kaggle constitue ainsi un cadre idéal pour entraîner rapidement des modèles complexes, notamment les LLMs, sans nécessiter de ressources matérielles locales.

3 Langage de programmation et bibliothèques

3.1 Python

Python est un langage de programmation open source, multi-plateformes, orienté objet et dynamique, reconnu pour sa syntaxe simple et sa facilité d'apprentissage. Il est largement utilisé dans des domaines variés tels que le développement web, la science des données, l'intelligence artificielle et le machine learning. Grâce à sa richesse en structures de données intégrées, sa flexibilité et la disponibilité de nombreuses bibliothèques puissantes, Python est aujourd'hui l'un des langages les plus utilisés, aussi bien par les débutants que par les experts en développement logiciel.

3.2 Bibliothèques

Parmi les bibliothèques de Python que nous avons utilisé dans notre implémentation :

- **Pandas :** Pandas est une bibliothèque open-source de Python qui fournit des structures de données rapides, flexibles et expressives, conçues pour faciliter la manipulation de données relationnelles ou étiquetées. Elle est particulièrement adaptée au traitement de données tabulaires provenant de fichiers CSV. Pandas offre de nombreuses fonctionnalités pour le nettoyage, la transformation et la visualisation des données. Très utilisée en science des données, cette bibliothèque permet une productivité élevée grâce à ses performances optimisées [40]
- **re (regular expressions)** est une bibliothèque standard de Python qui permet la recherche, le filtrage et la manipulation de chaînes de caractères à l'aide d'expressions régulières. Elle est largement utilisée pour l'extraction, le nettoyage et le traitement de données textuelles [41].
- **Scikit-learn :** Scikit-learn est une bibliothèque Python populaire et robuste pour machine learning. Elle fournit des outils efficaces pour la séparation des données en ensembles, l'évaluation des performances des modèles à l'aide de métriques variées, ainsi que pour leur validation. Ces fonctionnalités en font un outil essentiel pour la construction et l'analyse de modèles prédictifs.
- **Hugging Face :** Hugging Face est une plateforme open-source spécialisée dans le domaine du traitement du langage naturel (NLP). Elle offre une large collection de modèles pré-entraînés et de jeux de données, ainsi que des outils facilitant l'accès à des architectures avancées telles que BERT, GPT, et bien d'autres. Elle joue un rôle central dans l'intelligence artificielle appliquée au langage.
- **Transformers :** Transformers est une bibliothèque open-source développée par Hugging Face, conçue pour l'entraînement et l'inférence de modèles pré-entraînés basés sur l'architecture Transformer. Elle prend en charge le traitement du texte, de l'audio, de l'image et des données multimodales. Elle permet de fine-tuner facilement des modèles pour différentes tâches NLP comme la classification des sentiments [42].

4 Implémentation

4.1 Préparation de données

Les données utilisées dans cette étude proviennent de la combinaison de deux jeux de données publics disponibles sur GitHub [43] et sur la plateforme Mendeley Data [44].

Après avoir fixé notre objectif sur la classification binaire des sentiments (positif ou négatif), nous avons constitué un corpus contenant 84 765 commentaires.

Pour visualiser la répartition des sentiments dans ce jeu de données, nous avons utilisé le code présenté dans la Figure 3.1, qui trace la distribution des classes sous forme de graphique.

```
import matplotlib.pyplot as plt
import seaborn as sns

# Tracer la distribution des sentiments
ax = sns.countplot(data=df, x='label', palette=['#D00003', '#007703'])

# Remplacer les étiquettes de l'axe des x : 0 → Négatif, 1 → Positif
ax.set_xticklabels(['Négatif (0)', 'Positif (1)'])

# Ajouter les valeurs numériques au-dessus de chaque barre
for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}',
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center',
                fontsize=12, color='black',
                xytext=(0, 5), textcoords='offset points')


plt.title("Distribution des sentiments")
plt.savefig("Distribution des sentiments", dpi=600)
plt.show()
```

Figure 3. 1. Distribution des sentiments dans le jeu de données

- Nous avons également calculé le nombre de mots (tokens) dans chaque commentaire afin d'analyser la longueur des textes. Cette analyse nous a permis d'identifier les

commentaires trop courts ou trop longs. Enfin, nous avons représenté la distribution de ces longueurs à l'aide d'un histogramme. Le code correspondant à cette étape est illustré dans la figure 3.2 ci-dessous.

```
# Nombre de tokens par commentaire
df['token_count' ] = df['text' ]. apply (lambda x: len(str(x) .split()))

# Tracer la distribution du nombre de tokens par commentaire
plt.figure(figsize=(10, 6))
sns. histplot(df['token_count'], color='maroon')

# Ajouter les etiquettes des axes et le titre du graphique
plt.title("Distribution du nombre de tokens")
plt.xlabel("Nombre de tokens")
plt.ylabel("Fréquence")
plt.savefig("Distribution du nombre de tokens (avant suppression)", dpi=600)
plt.show()
```

Figure 3. 2. Distribution du nombre de tokens par commentaire (avant suppression).

4.2 Prétraitement de données

Dans cette phase, plusieurs étapes ont été utilisées pour nettoyer notre jeu de données, comme détaillé dans la sous-section 3.2 du deuxième chapitre. L'implémentation correspondante est présentée dans la figure 3.3.

```python
# Supprimer les phrases contenant moins de 4 tokens ou plus de 128 tokens
df = df[(df.token_count ⩽ 128) & (df.token_count ⩾ 4)]

# Supprimer les phrases non arabes
df = df[df.language == 'ar']

# Supprimer les doublons basés sur le texte
df = df.drop_duplicates(subset='text')

def preprocess_text(text):
    text = str(text)

    # 1. Supprimer les balises HTML
    text = re.sub(r'<.*?>', '', text)

    # 2. Supprimer les emojis
    text = emoji.replace_emoji(text, replace='')

    # 3. Normaliser les caractères arabes
    text = re.sub(r'[إأآا]', 'ا', text)
    text = re.sub(r'ه' ,'ة', text)
    text = re.sub(r'[ىئؤ]', 'ء', text)

    # 4. Supprimer les chiffres
    text = re.sub(r'\d+', '', text)

    # 5. Supprimer la ponctuation
    text = re.sub(r'[^\w\s]', '', text)

    # 6. Supprimer les mots consécutifs dupliqués
    words = text.split()
    text = ' '.join([words[i] for i in range(len(words)) if i == 0 or words[i] ≠ words[i - 1]])

    # 7. Normaliser les espaces
    text = ' '.join(text.split())

    return text
```

Figure 3. 3. Prétraitement de données

- **Division du jeu de données :** Après le nettoyage de notre jeu de données, nous l'avons divisé en trois sous-ensembles : entraînement 70 %, validation 15 % et test 15 %. Cette division a été réalisée tout en préservant la répartition des classes (positif, négatif) dans chaque sous-ensemble. Le code utilisé pour cette étape est présenté à la figure 3.4.

```python
from sklearn.model_selection import train_test_split

# Définir la colonne des étiquettes (labels)
label_column = 'label'

# Première séparation : 70 % pour l'entraînement, 30 % (validation + test)
train_df, temp_df = train_test_split(
    df,
    test_size=0.30,
    stratify=df[label_column],
    random_state=42
)

# Deuxième séparation : 15 % pour la validation, 15 % pour le test (à partir des 30 % restants)
val_df, test_df = train_test_split(
    temp_df,
    test_size=0.50,
    stratify=temp_df[label_column],
    random_state=42
)
```

Figure 3. 4. Division du jeu de données

4.3 Tokenisation

L'étape de tokenisation consiste à convertir les textes nettoyés en une représentation numérique compréhensible par les modèles de langage pré-entraînés (LLMs). Pour cela, nous avons utilisé le tokenizer associé à chaque modèle (AraBERT, CAMeLBERT, QARiB, mBERT, XLM-R, DistilBERT) disponible via la bibliothèque Hugging Face. La tokenisation a été appliquée sur les trois ensembles de données.

Pour éviter les problèmes liés à la variabilité de la longueur des séquences, nous avons utilisé un DataCollatorWithPadding, qui applique automatiquement un remplissage dynamique (dynamic padding) en fonction de la longueur maximale présente dans chaque lot de données (batch) lors de l'entraînement.

La figure 3.5 ci-dessous illustre l'ensemble du processus de tokenisation chargement de model.

```
from transformers import AutoTokenizer, AutoModelForSequenceClassification
from transformers import DataCollatorWithPadding
from datasets import Dataset

# Charger le tokenizer du modèle pré-entraîné

model_name = 'ahmedabdelali/bert-base-qarib'
#model_name = 'distilbert-base-multilingual-cased'
#model_name = 'bert-base-multilingual-cased'
#model_name = 'CAMeL-Lab/bert-base-arabic-camelbert-da'
#model_name = 'xlm-roberta-base'
#model_name = 'aubmindlab/bert-base-arabert'

tokenizer = AutoTokenizer.from_pretrained(model_name)

# Définir la fonction de tokenisation
def tokenize_function(examples):
        return tokenizer(
                examples["clean_text"],
                padding=False,
                truncation=True,
                max_length=128
        )

# Convertir les DataFrames pandas en datasets Hugging Face
train_dataset = Dataset.from_pandas(train_df[['clean_text', 'label']])
val_dataset = Dataset.from_pandas(val_df[['clean_text', 'label']])
test_dataset = Dataset.from_pandas(test_df[['clean_text', 'label']])

# Appliquer la tokenisation
train_dataset = train_dataset.map(tokenize_function, batched=True)
val_dataset = val_dataset.map(tokenize_function, batched=True)
test_dataset = test_dataset.map(tokenize_function, batched=True)

# Charger le modèle de classification
model = AutoModelForSequenceClassification.from_pretrained(
        model_name,
        num_labels=2  # Classification binaire
)

# padding dynamique
data_collator = DataCollatorWithPadding(tokenizer=tokenizer)
```

Figure 3. 5. Chargement du tokenizer, du modèle et du collateur de données.

4.4 Fine tuning

Le fine-tuning constitue une étape essentielle visant à adapter les modèles pré-entraînés (QARiB, mBERT, etc.) à notre jeu de données. Pour ce faire, nous avons défini les paramètres d'apprentissage nécessaires (taux d'apprentissage, nombre d'époques, taille des batchs, etc.) et lancé l'entraînement. La Figure 3.6 ci-dessous illustre ce processus.

```
compute_metrifrom transformers import TrainingArguments

training_args = TrainingArguments(
    output_dir="./sentiment_model_qarib",
    eval_strategy="epoch",                  # Évaluation à chaque époque
    save_strategy="epoch",                  # Sauvegarde du modèle à chaque époque
    learning_rate=2e-5,                     # Taux d'apprentissage
    per_device_train_batch_size=32,         # Batch size pour l'entraînement
    per_device_eval_batch_size=32,          # Batch size pour l'évaluation
    num_train_epochs=3,                     # Nombre total d'époques
    load_best_model_at_end=True,            # Charger le meilleur modèle à la fin
    metric_for_best_model="accuracy"        # Métrique utilisée pour choisir le meilleur modèle
)

from transformers import Trainer

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,
    tokenizer=tokenizer,
    data_collator=data_collator,
    compute_metrics=compute_metrics
)

# Lancement de l'entraînement
trainer.train()
```

Figure 3. 6. Fine-tuning.

4.5 Evaluation

Après l'entraînement des modèles, nous avons procédé à leur évaluation sur l'ensemble de test. Cette étape permet de mesurer la performance finale des modèles. L'implémentation de cette phase est illustrée dans la Figure 3.7 ci-dessous.

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Prédiction sur le jeu de test
predictions = trainer.predict(test_dataset)

# Conversion des logits en classes prédites
y_pred = np.argmax(predictions.predictions, axis=1)

# Récupération des vraies étiquettes
y_true = np.array(test_dataset['label'])

# Calcul des métriques
acc = accuracy_score(y_true, y_pred)
pre = precision_score(y_true, y_pred)
rec = recall_score(y_true, y_pred)
f1 = f1_score(y_true, y_pred)
```

Figure 3. 7. Evaluation

4.6 Démploiement

Afin de permettre l'utilisation pratique du modèle ayant obtenu les meilleures performances (QARiB), nous avons développé une interface web simple et accessible. Cette interface repose sur du HTML et du CSS personnalisé pour le frontend, tandis que le framework Flask en Python est utilisé pour gérer le backend (Figure 3.8). L'utilisateur peut saisir un commentaire en dialecte algérien. L'application traite ensuite ce texte à l'aide du modèle , puis affiche la prédiction du sentiment (positif ou négatif), accompagnée d'un score de confiance.

```python
from flask import Flask, request, render_template
from transformers import AutoTokenizer, AutoModelForSequenceClassification
import torch
import os

app = Flask(__name__)

# Chargement du modèle QARiB fine-tuné
model_path = "./model_qarib"
tokenizer = AutoTokenizer.from_pretrained(model_path)
model = AutoModelForSequenceClassification.from_pretrained(model_path)

@app.route("/", methods=["GET"])
def index():
    return render_template("index.html")

@app.route("/get_sentiment", methods=["POST"])
def get_sentiment():
    text = request.form["text"]

    # Préparation des données pour le modèle
    inputs = tokenizer(text, return_tensors="pt", truncation=True)
    with torch.no_grad():
        outputs = model(**inputs)
        probs = torch.nn.functional.softmax(outputs.logits, dim=-1)
        predicted_class = torch.argmax(probs, dim=1).item()
        confidence = round(probs[0][predicted_class].item(), 2)

    # Nom de la classe
    labels = ["Negative", "Positive"]
    prediction = labels[predicted_class]

    return render_template("index.html", prediction=prediction, text=text, confidence=confidence)

if __name__ == "__main__":
    app.run(debug=True)
```

Figure 3. 8. Déploiement

5 Résultats et discussion

5.1 Jeu de données avant le prétraitement

- **Jeu de données :** Notre jeu de données comprend 84 766 commentaires, répartis en deux colonnes : le texte du commentaire et sa classe de sentiment (positive ou négative). Le *wordcloud* présenté à la Figure 3.9, met en évidence les mots les plus fréquemment utilisés dans l'ensemble du corpus.

Figure 3. 9. Visualisation des mots les plus fréquents dans le corpus

Distribution de sentiment : Avant le prétraitement, les 84 766 commentaires étaient équitablement répartis entre les classes positives et négatives, comme illustré à la Figure 3.10. Cette distribution équilibrée est favorable à l'apprentissage du modèle, car elle permet de limiter les biais induits par un déséquilibre des classes, ce qui peut contribuer à de meilleures performances globales.

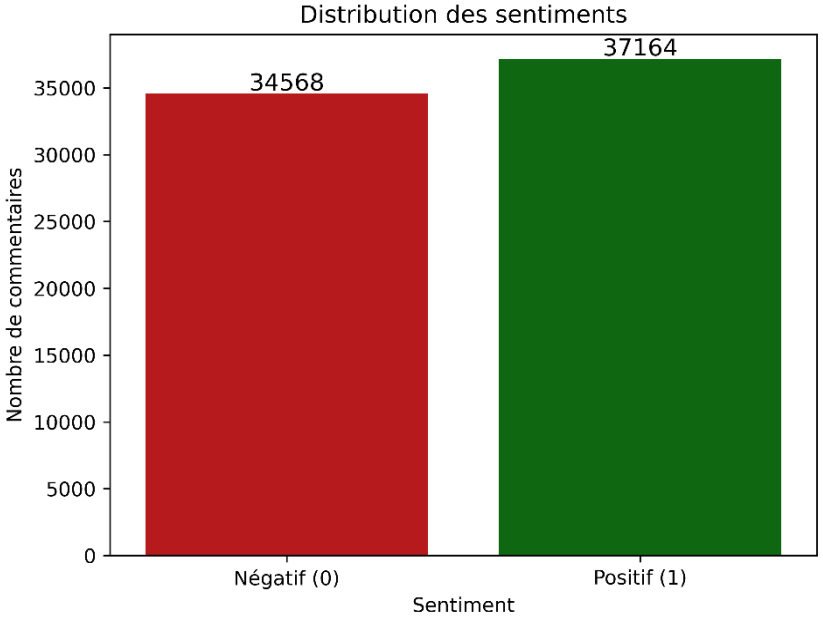

Figure 3. 10. Distribution des sentiments avant le prétraitement de données

- **Distribution de la longueur des commentaires** : Après analyse de notre jeu de données, nous avons constaté que la majorité des commentaires contiennent entre 4 et 128 mots (voir figure 3.11). C'est pourquoi, lors de la phase de prétraitement, nous avons choisi d'exclure les commentaires trop courts (moins de 4 mots) ou trop longs (plus de 128 mots), afin d'assurer une meilleure qualité d'entraînement.

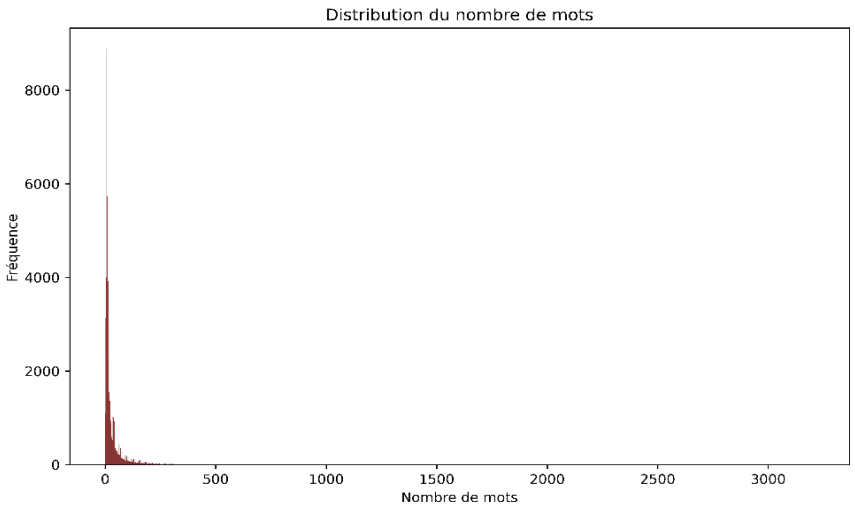

Figure 3. 11. Distribution du nombre de tokens (avant suppression)

- **Distribution de la langue :** Dans notre jeu de données, nous avons constaté que la grande majorité des commentaires (94 %) sont rédigés en arabe, tandis qu'une faible proportion est exprimée dans d'autres langues. Cette répartition est illustrée dans la Figure 3.12. Par conséquent, nous avons choisi de concentrer notre étude exclusivement sur les commentaires en langue arabe.

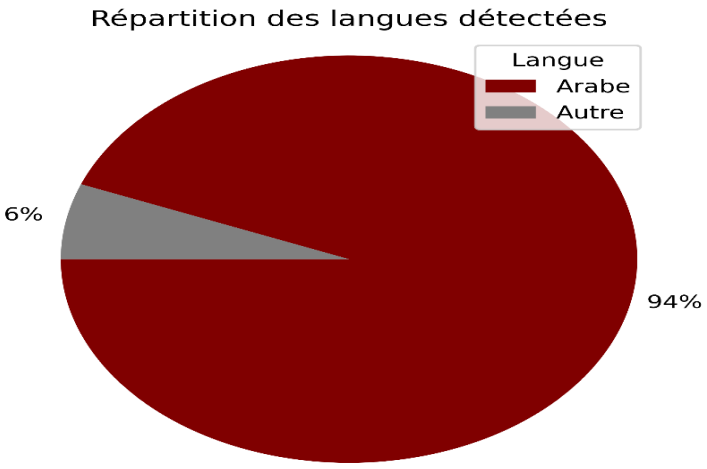

Figure 3. 12. Distribution des langues

Un échantillon de commentaires issus de notre jeu de données avant le nettoyage est présenté dans le Tableau 3.1, accompagné de leurs annotations de sentiment respectives. Cet échantillon illustre la nature brute de notre donnée.

Commentaire	**Label**
المطار يشهد على القرون الوسطى الفقر .آخر دولة تجلي الجالية والأغرب في ذلك يتكلمون الفرنسية	0
استاذ نورالدين انا باك 2021🙂 والله ربي يحفظك وربي يعطيك الصحة 🙂وربي يرحم الوالدة الكريمة💔💔🌹🖤	1
نذير هايل 😂مشاء الله مي سمحولي الغنا لي حطوه ماقاسوهاش كامل تبهديله كبيرة 😍	0
رحتو لروس باش تبنيوها علي راص الشعب الجزائري يا بني صهيون روحو اشرو يطبعولكم سلاح 🔫الخردة ديالهم وياكلولكم دراهم الشعب😏 حسبنا الله 🤲🙏 ونعم الوكيل	0
DZاميرنا ربي يخليك ديما بخير🌷👍👌✌👏	1
حسبي الله ونعم الوكيل في كل شخص مؤذي لأخيه 😖😔	0
باك 2020 ياااارب ❤️ان شاااء الله ننجحوو يااارب	1
واش هذا كرهتوني في اللغة الفرنسية عند بالي قناة جزائرية المهم على سلامتكم وخلاص 🤔🤔🤔	0
الله لاتربحكم يا مديري ووزير الصحة وكل قطاع الله يخرجها فيكم وفي صحتكم يارب يارب يعطيكم مرض ليخليكم ترشاو باه تحسو بالمرضى😭😭😭	0

Tableau 3. 1. Exemples de commentaires avant nettoyage

5.2 Jeu de données après le prétraitement

Après le prétraitement, nous avons obtenu un jeu de données composé de 67 335 commentaires, répartis comme illustré à la Figure 3.13. Cette répartition montre que l'équilibre de la distribution des sentiments a été préservé après le prétraitement.

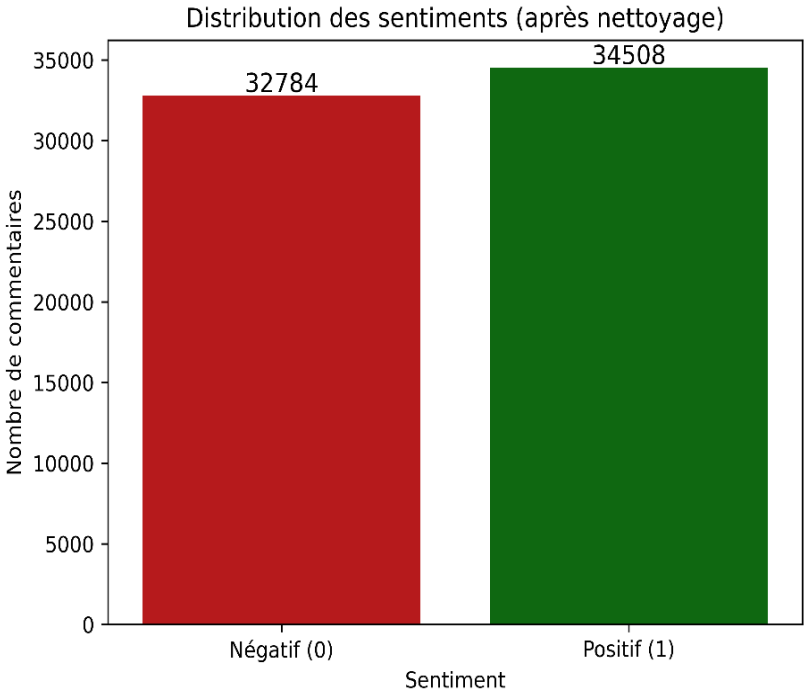

Figure 3. 13. Distribution de sentiment après le prétraitement

Le Tableau 3.2 ci-dessous présente la répartition des commentaires par catégorie de sentiment (négatif, positif) entre les ensembles d'entraînement, de validation et de test.

Sentiment	**Entraînement**	**Validation**	**Test**	**Total**
Négatif	22 940	4 922	4 922	32784
Positif	24 164	5 172	5 172	34508
Total	47 104	10 094	10 094	67 292

Tableau 3. 2. Répartition des données (entraînement, validation, test)

Dans le Tableau 3.3, nous présentons des exemples de commentaires avant et après le nettoyage. Il met en évidence l'importance du prétraitement de jeu de données

Avant prétraitement	**Après prétraitement**
المطار يشهد على القرون الوسطى الفقر .آخر دولة تجلي الجالية والأغرب في ذلك يتكلمون الفرنسية	المطار يشهد على القرون الوسطى الفقر اخر دوله تجلي الجاليه والاغرب في ذلك يتكلمون الفرنسيه
استاذ نورالدين انا باك 2021🙂 والله ربي يحفظك وربي يعطيك الصحة 🙂وربي يرحم الوالدة الكريمة💔💔🌹🖤	استاذ نورالدين انا باك والله ربي يحفظك وربي يعطيك الصحه وربي يرحم الواله الكريمه
نذير هايل 😂😍مشاء الله مي سمحولي الغنا لي حطوه ماقاسوهاش كامل تبهديله كبيرة	نذير هايل مشاء الله مي سمحولي الغنا لي حطوه ماقاسوهاش كامل تبهديله كبيره
رحتو لروس باش تبنيوها علي راص الشعب الجزائري يا بني صهيون روحو اشرو يطبعولكم سلاح الخردة ديالهم🔫 وياكلولكم دراهم الشعب😉 حسبنا الله 🤲🙏 ونعم الوكيل	رحتو لروس باش تبنيوها علي راص الشعب الجزاءري يا بني صهيون روحو اشرو يطبعولكم سلاح الخرده ديالهم وياكلولكم دراهم الشعب
امبرنا ربي يخليك ديما بخيرDZ🌷🌷👍✌👏	امبرنا ربي يخليك ديما بخير
حسبي الله ونعم الوكيل في كل شخص مؤذي لأخيه 😣😔	حسبي الله ونعم الوكيل في كل شخص مءذي لاخيه
باك 2020 ياااارب ❤️ان شاااء الله ننجحوو يااارب	ان شاااء الله ننجحوو يااارب باك يااارب
واش هذا كرهتوني في اللغة الفرنسية عند بالي قناة جزائرية المهم على سلامتكم وخلاص 😕😕😕	واش هذا كرهتوني في اللغه الفرنسيه عند بالي قناه جزاءريه المهم على سلامتكم وخلاص
الله لاتربحكم يا مديري ووزير الصحة وكل قطاع الله يخرجها فيكم وفي صحتكم يارب يارب يعطيكم مرض ليخليكم ترشاو باه تحسو بالمرضى😭😭😭	الله لاتربحكم يا مديري ووزير الصحه وكل قطاع الله يخرجها فيكم وفي صحتكم يارب يعطيكم مرض ليخليكم ترشاو باه تحسو بالمرضى

Tableau 3. 3.. Exemples de commentaires avant et après le nettoyage

5.3 Fine-tuning

Le tableau 3.4 présente les résultats de fine tuning les modèle LLM spécifiquement sur des corpus arabes offrent les meilleures performances dans notre tâche d'analyse de sentiments en dialecte algérien. Le modèle QARiB se distingue comme le plus performant avec un F1-score de 91.3% ce qui indique une très bonne capacité à équilibrer entre précision (91.2%) et rappel (91.5%). CamelBERT obtient également d'excellents résultats avec une précision de 92%, et F1 score de 90.7%, et aussi une Accuracy de 90.6% qui démontrant sa capacité à minimiser les

faux positifs tout en maintenant un bon équilibre global. AraBERT, bien que légèrement en dessous des deux modèles précédents, reste très compétitif avec un bon rappel de 90.9 %, un F1-score de 89.3 % et une Accuracy de 88.9 %.

Modèle	Accuracy	Précision	Rappel	F1-score
QARiB	0.911	0.912	0.915	0.913
CamelBERT	0.906	0.920	0.894	0.907
AraBERT	0.889	0.878	0.909	0.893

Tableau 3. 4. Fine-Tuning des Modèles LLM spécialisés en arabe

D'autre part, le Tableau 3.5 présente les performances des modèles multilingues. Parmi eux, XLM-R affiche les meilleures performances avec un F1-score de 89.4% avec une bonne cohérence entre les différentes métriques, Cela montre sa capacité à généraliser même sur des textes dialectaux. Le modèle mBERT affiche une précision élevée (90.1%) mais un rappel plus faible (85.8%). Enfin, DistilBERT, bien qu'étant un modèle plus léger, reste efficace malgré des performances inférieures, avec un F1-score de 87.1% et une Accuracy de 87%. Globalement, bien que les modèles multilingues soient utiles, ils sont surpassés par les modèles spécialisés pour la langue arabe dans cette tâche.

Modèle	Accuracy	Précision	Rappel	F1-score
XLM-R	0.892	0.896	0.892	0.894
mBERT	0.879	0.901	0.858	0.879
DistilBERT	0.870	0.886	0.857	0.871

Tableau 3. 5. Fine-Tuning des Modèles LLM multilingues.

5.4 Déploiement

La figure 3.14 présente l'interface web développée afin de faciliter l'interaction avec l'utilisateur.

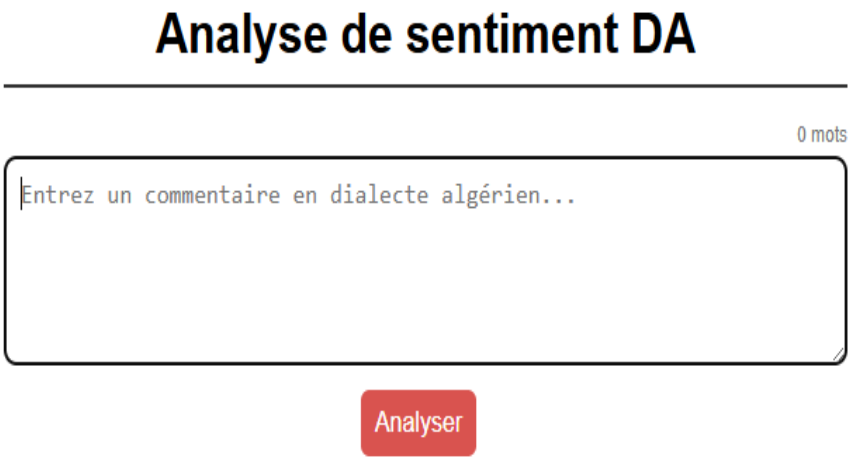

Figure 3. 14. Interface Web

La figure 3.15 illustre un exemple de test réalisé à l'aide de cette interface, permettant d'évaluer le modèle QARiB sur un commentaire en dialecte algérien.

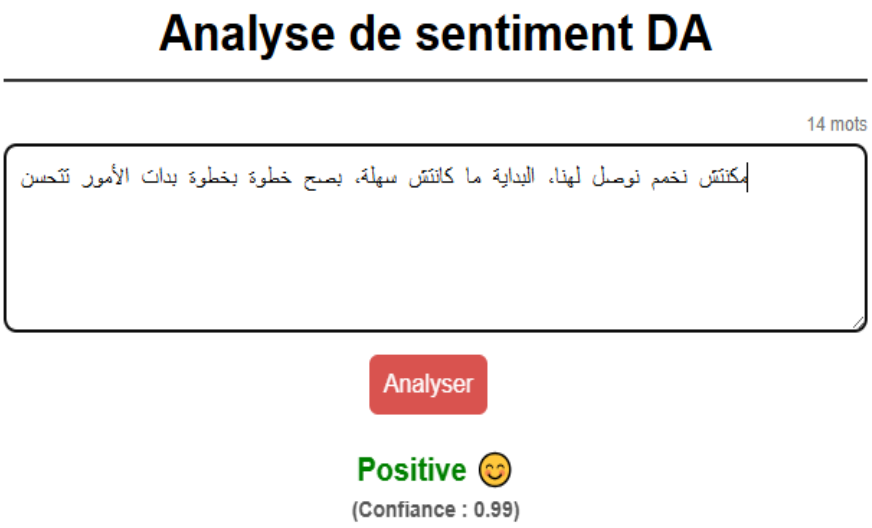

Figure 3. 15. Exemple de test

6 Expériences complémentaires

Afin d'évaluer la capacité de généralisation de nos modèles, nous avons testé leurs performances à l'aide de deux jeux de données : l'un en dialecte tunisien et l'autre en dialecte marocain (voir exemple dans le tablueau 3.6).

MSAC (Marocain)	TSAC (Tunisien)
أجمل وانظف الكلبات هي كلبات ديال اغاني سعد لمجرد	مكشر متغشش مافهمتوش لينات كحل بصراحة ماسط
كلنا خائنون	ماسط وساذج في تفكيرو ....كلمات تافهة من عبد أتفه
كبر مقتا عند الله ان تقولون مالا تفعلون.	بلاصتك في الحبس والا الحرق انت وامثالك حتى الموت
عذرا يا رسول الله فقد فرطنا فى أولي القبلتين وثالث الحرمين 😢	Wa7dek ya semi l fehri. Équipe 8ayr 3adyya w 3la rasshom ENA kerim l 8arbi. Min edho7k lin chra9t za3maaaa..... Yitwelleh rabbi yo9tok biddo7k hhhhh
اسأل الذي جمعنا في دنيا فانية أن يجمعنا ثانية في جنة قطوفها دانية عام هجري سعيد.	Ana nmout 3liha meriem ben chaabane 7louwa fi kol dawir
علينا وعليك بالصحة والعافية والتألق والمزيد من النجاح للوصول الى العالمية ولو انك قد وصلتها اتمنى لك المزيد	انا نحبها بارشا اااا ممثلة بارعة

Tableau 3. 6. Exemple de MSAC et TSAC

L'évaluation a été réalisée avant et après l'application des étapes de prétraitement, afin d'observer l'impact de cette étape sur les performances des modèles.

6.1 Évaluation sur le dialecte marocain

Le tableau 3.7 ci-dessous illustre les performances d'évaluation des différents modèles sur le corpus marocain, avant et après prétraitement.

Modèle	Avant prétraitement				Après le prétraitement			
	Accuracy	Précision (macro)	Rappel (macro)	F1 score (macro)	Accuracy	Précision (macro)	Recall (macro)	F1 score (macro)
QARiB	0.8910	0.89	0.89	0.89	0.8869	0.89	0.89	0.89
CamelBERT	0.8845	0.88	0.88	0.88	0.8791	0.88	0.88	0.88
AraBERT	0.8460	0.85	0.85	0.85	0.8555	0.85	0.86	0.86
XLM	0.8680	0.87	0.87	0.87	0.8124	0.81	0.81	0.81
mBERT	0.8170	0.82	0.82	0.82	0.8247	0.83	0.82	0.82
DistilBERT	0.7960	0.80	0.80	0.80	0.8124	0.81	0.81	0.81

Tableau 3. 7. Résultat d'évaluation MSAC avant et après le prétraitement

Avant prétraitement, les meilleurs résultats ont été obtenus par QARiB et CamelBERT avec une accuracy de 89.10% et 88.45 respectivement, confirmant son aptitude à capturer efficacement les caractéristiques linguistiques du dialecte marocain. XLM et AraBERT suivent de près avec des scores élevés, illustrant une bonne généralisation initiale.

Alors que après prétraitement les résultats montre que les performances des modèles restent relativement stables , ou QArib conserve son avantage avec une accuracy de 88.69 et F1 score macro de 89%, ce qui en fait le modèle le plus stable avant et après nettoyage, On observe également de légères améliorations d'accuracy pour certains modèles comme DistilBERT (+1.64 %), mBERT (+0.77 %) et AraBERT (+0.95 %), soulignant l'utilité du prétraitement pour épurer les données tout en maintenant les performances globales. Cependant, certains modèles comme Qarib et XLM enregistrent une très légère baisse de performance , probablement due à la réduction de la taille de jeu de données: avant prétraitement, le corpus contenait 2 000 exemples équilibrés (1 000 par classe), tandis qu'après nettoyage, il est réduit à 1 786 exemples, avec un léger déséquilibre entre les classes (945 négatifs et 841 positifs). Malgré ce déséquilibre, les scores sont restés globalement stables, ce qui témoigne de la robustesse des modèles évalués.

6.2 Évaluation sur le dialecte tunisien

Le tableau 3.8 ci-dessous présente les performances des modèles évalués sur le corpus tunisien, avant et après prétraitement.

Modèle	Avant prétraitement				Après le prétraitement			
	Accuracy	Précision (macro)	Rappel (macro)	F1 score (macro)	Accuracy	Précision (macro)	Recall (macro)	F1 score (macro)
QARiB	0.7765	0.78	0.77	0.77	0.9782	0.98	0.98	0.98
CamelBERT	0.7449	0.80	0.84	0.83	0.9716	0.97	0.97	0.97
AraBERT	0.7525	0.80	0.74	0.74	0.9546	0.95	0.95	0.95
XLM	0.7575	0.78	0.77	0.77	0.9420	0.94	0.94	0.94
mBERT	0.7343	0.76	0.74	0.73	0.9424	0.95	0.94	0.94
DistilBERT	0.7150	0.76	0.72	0.71	0.9366	0.94	0.93	0.94

Tableau 3. 8. Résultat d'évaluation TSAC avant et après le prétraitement

Avant prétraitement, les résultats révèlent déjà une certaine capacité de généralisation des modèles, bien que les performances restent globalement inférieures à celles observées pour le dialecte marocain. Cette diminution des performances s'explique en partie par la composition linguistique du jeu de données tunisien, qui contient un pourcentage important de commentaires rédigés dans des langues autres que l'arabe (voir Figure 1.3). Cela réduit la proportion de textes exploitables en dialecte arabe pour l'apprentissage des modèles. Malgré ce contexte, certains modèles comme CamelBERT (F1-score : 83 %), suivi de XLM-R et QARiB (77 %), démontrent une robustesse notable.

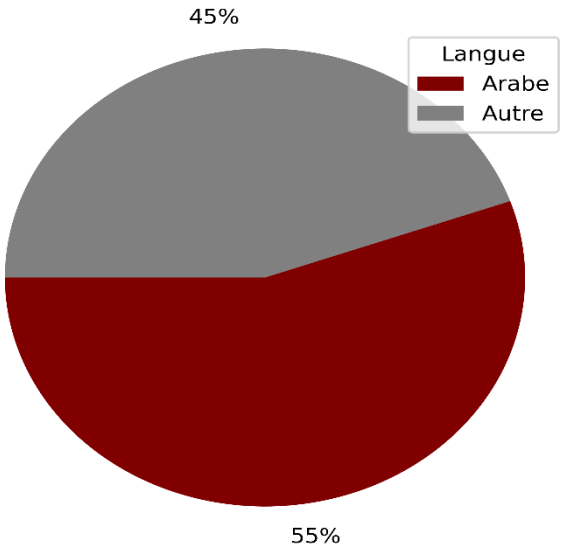

Figure 3. 16. Répartition des langues détectées du TSAC

Après prétraitement, les résultats sont nettement améliorés pour l'ensemble des modèles testés, avec des avancées allant jusqu'à 20 points pour certains pour certains d'entre eux. On observe, par exemple, une progression significative pour QARiB, dont l'accuracy passe de 77.65 % à 97.82 %, et pour CamelBERT, qui évolue de 74.49 % à 97.16 %. Bien que la taille du jeu de données ait été réduite de 17 065 à 4 999 commentaires après prétraitement, l'ensemble des modèles bénéficie d'une amélioration notable sur toutes les métriques.

Cette progression remarquable est principalement due à l'efficacité des étapes de prétraitement mises en œuvre. Celles-ci ont permis de filtrer les données non arabes, d'éliminer les commentaires bruités et d'améliorer la cohérence linguistique du corpus. En réduisant la variabilité linguistique, les modèles ont pu mieux généraliser et produire des résultats significativement plus fiables.

7 Conclusion

Dans ce chapitre, nous avons présenté une vue d'ensemble de la mise en œuvre du système, en détaillant l'environnement de travail, le langage de programmation et les bibliothèques utilisées. Des extraits de code illustratifs ont également été fournis afin de clarifier chaque étape du processus. Enfin, une analyse approfondie des résultats obtenus a été proposée.

Conclusion générale

L'usage des réseaux sociaux est devenu une activité quotidienne incontournable dans la société actuelle. Leur utilisation massive génère une quantité considérable de données textuelles, souvent spontanées et non structurées, qui renferment un volume important d'informations exploitables. Ces informations peuvent être extraites et mises à profit dans plusieurs tâches du traitement automatique du langage naturel, notamment l'analyse des sentiments, qui vise à identifier automatiquement l'opinion exprimée dans un texte.

Dans ce mémoire, nous nous sommes intéressés à l'analyse des sentiments en dialecte algérien, en nous concentrant sur une classification binaire (positive vs négative). Notre objectif principal était d'adapter des modèles de langage pré-entraînés (LLMs) à cette tâche, en tenant compte des spécificités linguistiques du dialecte algérien, telles que l'absence de norme, la variation régionale et l'alternance linguistique fréquente.

Pour atteindre cet objectif, nous avons mis en place une méthodologie rigoureuse comprenant la collecte et le nettoyage de données issues de corpus existants, la tokenisation, la représentation vectorielle des textes, puis le fine-tuning de plusieurs modèles pré-entraînés, spécialisés en arabe (AraBERT, CAMeLBERT, QARiB) ou multilingues (mBERT, XLM-R, DistilBERT), suivi d'une évaluation de leurs performances.

Les résultats obtenus ont montré que le modèle QARiB offre les meilleures performances sur notre tâche, surpassant les autres modèles avec une accuracy de 91,1 % et un F1 score de 91,3 %. Nous avons également évalué la capacité de généralisation de ces modèles sur deux autres dialectes arabes du Maghreb (marocain et tunisien). Les résultats ont révélé une amélioration notable après prétraitement, en particulier grâce à l'élimination des commentaires non arabes et bruités.

Cependant, bien que les résultats soient encourageants, certaines contraintes subsistent, notamment la diversité linguistique interne du dialecte algérien, le manque de ressources annotées de qualité et les difficultés liées à l'écriture non standardisée (utilisation de l'arabizi, alternance avec le français, etc.), ce qui complique l'apprentissage des modèles.

Pour aller plus loin, plusieurs pistes peuvent être explorées :

- Développer des modèles capables de gérer simultanément plusieurs dialectes proches (algérien, tunisien, marocain) de manière robuste.
- Exploiter les ressources en arabe standard (MSA) et en français pour améliorer les performances en dialecte algérien.
- Créer des corpus ouverts spécifiques au dialecte algérien, notamment à partir de sources sociales variées.

Ces perspectives contribueraient à renforcer les capacités des systèmes de traitement du langage naturel à mieux comprendre et traiter la richesse linguistique des dialectes arabes, et en particulier celui de l'Algérie.

Bibliographie

[1] Versteegh, K. (2014). *Arabic Language.* Edinburgh: Edinburgh University Press.

[2] Mashaabi, M., Al-Khalifa, S., & Al-Khalifa, H. (2024). A Survey of Large Language Models for Arabic Language and its Dialects. *arXiv preprint arXiv:2410.20238*.

[3] Kirchhoff, k., Bilmes, J., Das, S., Duta, N., Egan, M., & Ji, G. (2003). Novel approaches to Arabic speech recognition: Report from the 2002 Johns-Hopkins summer workshop. *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03).* IEEE.

[4] Harrat , S., Meftouh, K., Abbas, M., Smaïli, K., & Hidouci, W.-K. (2016). An Algerian dialect: Study and resources. *International Journal of Advanced Computer Science and Applications, 7*, 384–396.

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

[6] Moudjari, L., & Akli-Astouati, K. (2019, January). Construction et exploitation d'un corpus multilingue algérien pour l'analyse des opinions et des émotions. *EGC*, (pp. 321–326).

[7] Touileb, S. (2022). NERDz: a preliminary dataset of named entities for Algerian. *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, (pp. 95–101).

[8] Dahou , A., Cheragui, M., & Abdelali, A. (2023, September). Performance Analysis of Arabic Pre-Trained Models on Named Entity Recognition Task. *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, (pp. 458–467).

[9] Boulesnane, A., Saidi, Y., Kamel, O., Bouhamed, M., & Mennour, R. (s.d.). DZchatbot: A Medical Assistant Chatbot in the Algerian Arabic Dialect using Seq2Seq Model.

[10] Alimi, T., Boujebane, R., Derouich, W., & Belguith, L. (2024). Fine-Tuned Transformers for Translating Multi-Dialect Texts to Modern Standard Arabic.

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

[12] Wankhade, M., Rao, A., & Kulkarni, C. (2022). A survey on sentiment analysis methods, applications, and challenges.

[13] Gupta, S., Ranjan, R., & Singh, S. (2024). Comprehensive study on sentiment analysis: From rule-based to modern LLM based system. *arXiv preprint arXiv:2409.09989*.

[14] Madhoushi, Z., Hamdan, A., & Zainudin, S. (2015). Sentiment Analysis Techniques in Recent Works.

[15] Rawal, G., Rawal, R., & Shah, H. (2019). A comparative study between artificial neural networks and conventional classifiers for predicting diagnosis of breast cancer.

[16] VAJJALA, S., MAJUMDER, B., GUPTA, A., & al. (2020). Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems. *O'Reilly Media* .

[17] Vaswani, A., Shezeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30.*

[18] MATAOUI, M., ZELMATI, O., & BOUMECHACHE, M. (2016). A Proposed Lexicon-Based Sentiment Analysis Approach for the Vernacular Algerian Arabic.

[19] GUELLIL, I., ADEEL, A., AZOUAOU, F., & al. (2018). SentiALG: Automated Corpus Annotation for Algerian Sentiment Analysis. *arXiv preprint arXiv:1808.05079*.

[20] Abdelli, A., Guerrouf, F., Tibermacine, O., & Abdelli, B. (2019, December). Sentiment analysis of Arabic Algerian dialect using a supervised method. *2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS)*, (pp. 1-6).IEEE.

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

[22] Soumeur, A., Mokdadi, M., Guessoum, A., & Daoud, A. (2018). Sentiment analysis of users on social networks: overcoming the challenge of the loose usages of the Algerian Dialect. *Procedia Computer Science, 142*, (pp. 26-36). Récupéré sur https://doi.org/10.1016/j.procs.2018.10.458

[23] Mazari, A., & Djeffal, A. (2021). Deep learning-based sentiment analysis of Algerian dialect during Hirak 2019. *2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH)*, (pp. 233–236). IEEE.

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

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

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

[27] Abdeladeim, A., Dahou, A., Chergui, M., & Mathiak, B. (2024). FASSILA: A Corpus for Algerian Dialect Fake News Detection and Sentiment Analysis. *Procedia Computer Science, 244*, 397–407.

[28] REHAIEM, E., & DIDA, M. (2021). Algerian Dialect text clustering based on Emotion detection (Doctoral dissertation).

[29] Boujou, E., Chataoui, H., Mekki, A., Benjelloun, S., Chairi, I., & Berrada, I. (2021). An open access NLP dataset for Arabic dialects: Data collection, labeling, and model construction. *arXiv preprint arXiv:2102.11000*.

[30] Bousmaha, K., Hamadouche, K., Cheurfaoui, N., & Hadrich-Belguith, L. (2024). Subject Detection of Algerian Posts for Opinion Analysis. *Ingénierie des Systèmes d'Information, 29(3)*(821).

[31] Dahou, A., & Cheragui, M. (2023). Dzner: A large Algerian named entity recognition dataset. *Natural Language Processing Journal, 3*(100005).

[32] Antoun, W., Baly, F., & Hajj, H. (2021). TunBERT: Pretrained Contextualized Text Representation for Tunisian Dialect. *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

[33] Abdelali, A., Hassan, S., Mubarak, H., Darwish, K., & Samih, Y. (2021). Pre-training BERT on Arabic tweets: Practical considerations. *arXiv preprint arXiv:2102.10684*.

[34] Innoue, G., Alhafni, B., Bouamor, H., & Habash, N. (2021). The interplay of variant, size, and task type in Arabic pre-trained language models. *arXiv preprint arXiv:2103.06678*.

[35] Antoun, W., Baly, F., & Hajj, H. (2020). Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*.

[36] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019, June). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, (pp. 4171–4186).

[37] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

[38]. Oussous, A., Benjelloun, F., Lahcen, A., & Belfkih, S. (2020). ASA: A framework for Arabic sentiment analysis. *Journal of Information Science, 46*(4), 544-559.

[39] Mdhaffer, S., Bougares, F., Esteve, Y., & Hadrich-Belguith, L. (2017, April). Sentiment analysis of Tunisian dialects: Linguistic resources and experiments. *Proceedings of the Third Arabic Natural Language Processing Workshop (WANLP)*, (pp. 55-61).

[40] McKinney, W. (2010). Data structures for statistical computing in Python. *445*(1), 51-56.

Webographie

[41] Google Developers. *Regular expressions*. Consulté le 6 12, 2025, sur https://developers.google.com/edu/python/regular-expressions?hl=fr

[42] Hugging face. *Transformers documentation*. Consulté le 6 12, 2025, sur https://huggingface.co/docs/transformers/index

[43] Abdelli, A. *DzSentiA: Algerian dialect sentiment analysis dataset*. Consulté le 5 2,2025, sur GitHub: https://github.com/adelabdelli/DzSentiA

[44] Benmounah, Z. (2023). *Algerian Dialect*. Consulté le 15 3, 2025, sur Mendeley Data: https://doi.org/10.17632/zzwg3nnhsz.1