Democratic and Popular Republic of Algeria

Ministry of Higher Education and Scientific Research

University of Médéa / Faculty of Sciences

Department of Mathematics and Computer Science

Graduation Project

Submitted for graduation in **Master of Computer Science**

Option: System Engineering and Web Technology

Transfer Learning and Deep Learning for Multilingual Algerian Dialect Hate Speech Detection

Proposed and directed by:

Dr. Ahmed Cherif Mazari

Prepared and presented by:

Ms.Djamila Menifi Ms.Wiam Moussa

2021/2022

Acknowledgment

First of all, we thank my God Allah for the will and the patience he has given us. given to carry out this work and put us on the right path.

We'd also like to express our heartfelt gratitude to our Advisor Ahmed Cherif MAZARI, whose collective support, guidance, and experience helped us complete our thesis. We are grateful for his willingness to share his knowledge and enthusiasm with us as we embark on this adventure.

Also, all of our university lecturers that assisted us during our studies.

Finally, we want to express our gratitude to our family and friends for

Dedication

First of all, I dedicate this work to our almighty God,

To My dear parents ❤️ (Abdel kader/ Houria) AND dear husband ❤️ (Youcef), for their understanding and support morally and financially,

To My dear daughter Eline ❤️ (NINA)

To My lovely brothers and sisters ❤️, my eternal gratitude

To all my friends ❤️.

Djamila Menifi

Dedication

To my dear parents, Ali Moussa and Benouadeh hayet, who supported me emotionally with their blessings and prayers.

To my sisters mira,houdna and Rawan, MY FAMILLY,

And to all My brothers "Mohamed amine, youness, and Mohamed habib

To my partner (djamila),

And To all my dear friends, Houda, Asma, Kahina,Kania....

To all the promotion of 2nd year master's degree in

Computer Science (2021/2022). To everyone who knows me.

Wiam Moussa

Abstract

Social network have a great impact not only on information availability but also on its generation by enabling users to add their own contents to web without having any specific programming skills. These new communication services make human communication easier and people closer to each other.

Despite to their uncountable benefits, social media have their negative implications due to their misuse.

Abusive behaviors like Hate speech and all other shapes of toxicity have increasingly taken place in these communication means making of online environments more and more toxic.

The aim of this project is dedicated, first to hate speech detection using Deep learning and Transfer learning models applied on Algerian dialect written in multilingual language, second to create a new dataset related to this problematic.

In this study, we test several models, Models of machine learning With TF-IDF and With Word2Vec embedding (Randomforestclassifier / MultinomialNB / Logistic Regression / SGD Classifier/ SVC classifier), Deep learning models Without / With FastText (LSTM/ GRU/ Bi-LSTM/ Bi-GRU/ CNN), Transfer learning models (Distilbert-base-uncased/ Bert-base-multilingual-uncased/ Bert-base-arabic), Bert layer with GRU and LSTM (BERT embedding +LSTM/ BERT embedding +GRU/ BERT embedding+GRU and LSTM) and Ensemble learning to improve an automated detection of Algerian dialect written in multilingual language.
The best models is transfer learning (Bert layer with GRU)F1-Score 86 % accuracy 87% afetr Deep learning models (LSTM)F1-Score 87% accuracy 85%.

Keywords: hate speech, Algerian, machine learning, Word2Vec, Randomforestclassifier, MultinomialNB, Logistic Regression, SGD Classifier, SVC classifier, deep learning, FastText, LSTM, GRU, Bi-LSTM, Bi-GRU, CNN, Transfer learning, Distilbert-base-uncased, Bert-base-multilingual-uncased, Bert-base-arabic, Ensemble learning, text classification, NLP, Bert model.

ملخص

للشبكات الاجتماعية تأثير كبير ليس فقط على توافر المعلومات ولكن أيضًا على إنشائها من خلال تمكين المستخدمين من إضافة محتوياتهم الخاصة إلى الويب دون امتلاك أي مهارات برمجة محددة. خدمات الاتصال الجديدة هذه تجعل التواصل البشري أسهل ويقترب الناس من بعضهم البعض

على الرغم من فوائدها التي لا تُحصى ، إلا أن وسائل التواصل الاجتماعي لها آثارها السلبية بسبب سوء استخدامها مثل الكلام الذي يحض على الكراهية، مما يجعل البيئات عبر الإنترنت أكثر سمية.

الهدف من هذا المشروع مخصص ، أولاً لاكتشاف الكلام الذي يحض على الكراهية باستخدام نماذج التعلم العميق ونقل التعلم المطبقة على اللهجة الجزائرية المكتوبة بلغة متعددة اللغات ، والثاني لإنشاء مجموعة بيانات جديدة تتعلق بهذه المشكلة.

في هذه الدراسة ، قمنا باختبار العديد من النماذج ونماذج التعلم الآلي باستخدام TF-IDF ومع تضمين Word2Vec
(Randomforestclassifier / MultinomialNB/الانحدار اللوجستي / مصنف SGD / مصنف SVC)

نماذج التعلم العميق بدون / مع FastText (LSTM/ GRU/ Bi-LSTM/ Bi-GRU/ CNN) ، نماذج تعلم التحويل (Distilbert-base-uncased / Bert-base-multilingual-uncased / Bert-base-arabic) ، طبقة بيرت مع GRU و LSTM (تضمين BERT + تضمين LSTM / BERT + GRU / دمج BERT + GRU و LSTM) وتعلم المجموعة لتحسين الاكتشاف الآلي للللهجة الجزائرية المكتوبة بلغة متعددة اللغات.
أفضل النماذج هي نقل التعلم (طبقة بيرت مع F1 (GRU- نقاط 86٪ دقة 87٪ afetr نماذج التعلم العميق F1-Score 87) LSTM٪ دقة 85٪.

الكلمات الرئيسية: خطاب الكراهية ، الجزائري ، التعلم الآلي ، Word2Vec، Randomforestclassifier، MultinomialNB ، الانحدار اللوجستي ، مصنف SGD ، مصنف SVC ، التعلم العميق ، FastText، LSTM، GRU، Bi-LSTM، Bi-GRU، CNN ، نقل التعلم ، Distilbert-base غير محدد ، بيرت-قاعدة-متعدد اللغات-غير محدد ، بيرت-القاعدة-العربية ، مجموعة التعلم ، تصنيف النص ، البرمجة اللغوية العصبية ، نموذجBERT .

RESUME

Les réseaux sociaux ont un grand impact non seulement sur la disponibilité de l'information mais aussi sur sa génération en permettant aux utilisateurs d'ajouter leurs propres contenus sur le web sans avoir de compétences spécifiques en programmation. Ces nouveaux services de communication facilitent la communication humaine et rapprochent les gens les uns des autres.

Malgré leurs innombrables avantages, les médias sociaux ont des implications négatives en raison de leur mauvaise utilisation.

Des comportements abusifs comme le discours de haine et toutes les autres formes de toxicité ont de plus en plus pris place dans ces moyens de communication, rendant les environnements en ligne de plus en plus toxiques.

L'objectif de ce projet est dédié, d'une part à la détection des discours de haine à l'aide de modèles d'apprentissage profond et d'apprentissage par transfert appliqués au dialecte algérien écrit en langue multilingue, d'autre part à la création d'un nouvel ensemble de données lié à cette problématique.

Dans cette étude, nous testons plusieurs modèles, Modèles d'apprentissage automatique Avec TF-IDF et Avec embedding Word2Vec (Randomforestclassifier / MultinomialNB / Régression Logistique / SGD Classifier/ SVC classifier), Modèles d'apprentissage profond Sans / Avec FastText (LSTM/ GRU/ Bi- LSTM/ Bi-GRU/ CNN), Modèles d'apprentissage par transfert (Distilbert-base-uncased/ Bert-base-multilingual-uncased/ Bert-base-arabic), Bert layer avec GRU et LSTM (BERT embedding +LSTM/ BERT embedding +GRU / BERT embedding+GRU and LSTM) et Ensemble learning pour améliorer une détection automatisée du dialecte algérien écrit en langue multilingue.
Le meilleur modèle est l'apprentissage par transfert (couche Bert avec GRU) F1-Score 86 % précision 87 % après les modèles d'apprentissage en profondeur (LSTM) F1-Score 87 % précision 85 %.

Mots clés : discours de haine, Algérien, apprentissage automatique, Word2Vec, Randomforestclassifier, MultinomialNB, Régression logistique, SGD Classifier, SVC classifier, deep learning, FastText, LSTM, GRU, Bi-LSTM, Bi-GRU, CNN, Transfer learning, Distilbert-base -uncased, Bert-base-multilingual-uncased, Bert-base-arabic, Apprentissage d’ensemble, Classification de texte, NLP, modèle BERT.

Table of Contents

List of figures
List of tables
List of acronyms
Introduction

Chapter I: **Hate speech language Detection**

Introduction 3
1. Definitions and Terminology 3
1.1. Violence 3
1.2. Offensive Language 3
1.3. Hate speech 3
1.4 hate speech en internet 3
1.5. Bullying 4
2. Detrimental Effects of offensiveness 4
2.1. Violence against women 4
2.2. Violence against children 4
3.2. Real World Systems for Offensiveness Detection 5
4.2 Hate speech detection using automated methods: 6
5. Arabic language 7
7. Fighting hate speech in Algerian: 9
Conclusion 9

Chapter II: **Deep Learning and transfer learning techniques**

Part 1: Introduction to Deep Learning and Transfer learning 10
Introduction : 10
1. Machine learning : 10
1.1. Supervised learning 10
1.2. Unsupervised learning 10
1.3. Reinforcement (Semi-supervised) learning 11
1.4. Machine learning algorithms 11
1.5. Feature Engineering in Machine Learning 12
2. Deep Learning (DL) 13
2.1 The importance of deep learning 13
2.3. Learning Process of a Deep Neural Networks 14
2.3.1. Activation function 16

2.3.2. Backpropagation components 17
2.4.3. Loss function 17
2.3.4. Optimizer 18
2.3.5. Gradient descent 18
2.4. Deep Learning Performance 18
2.5. Popular Deep Learning Algorithms 19
2.5.1. Deep Neural Network 19
2.5.2. Recurrent Neural Networks (RNNs) 19
2.5.3. Long Short-Term Memory (LSTM) Network 20
2.5.5. Gated Recurrent Units (GRUs) 22
3. Transfer Learning (TL) 23
3.1. Transformer 23
3.1.1. Key, Value, and Query 24
3.1.2. Self-attention 24
3.1.3. Multi-head Attention 25
3.1.4. Encoder 26
3.1.5. Decoder 26
3.2. Recent Language Understanding Models Based on Transfer Learning 27
3.2.1. Embeddings from Language Models (ELMo) 27
3.2.2. Bidirectional Encoder Representation from Transformers (BERT) 27
A. Masked LM (MLM) 28
B. Next Sentence Prediction (NSP) 29
4. Ensemble Learning 30
5. Attention Layer 30
5.1 Attention Mechanism 30
5.1.1 A Family of Attention Mechanisms 32
5.1.2. Summary 32
5.1.3. Self-Attention 33
5.1.4. Soft vs Hard Attention 34
5.1.5. Global vs Local Attention 34
6. Embedded Vectors 34
6.1. Collobert and Weston (C&W) model 34
6.2. Word2Vec 34
6.3. FastText 35
6.5. Graphs from LOw-level unit Modeling (GLoMo) 35

6.6. Universal Language Model Fine-tuning (ULMFiT) 35
Part2: 36
1. Related work: 36
1.1. Current state in Hate Speech Detection and Related Concepts: 36
1.1.2. Abusive language detection 36
1.1.3. Cyber bullying detection 36
1.1.4 . Radicalization detection 37
1.1.5. Hate speech detection 37
1.2. Arabic Hate Speech Detection 37
1.2.1. Arabic anti-social behaviour detection 38
1.2.2. Arabic hate speech detection 38
Algerian dialect hate speech detection: 39
Summary and Analysis 39
Conclusion 46
Chapter III: **Proposed approach for Algerian hate speech detection**
Introduction 47
1. Global Architecture 47
2. Methodology detection hate speech 48
2.1. Data creation: 49
2.2. Data collection: 49
2.3. Comments filtering : 51
2.4. Comments annotation 53
2.5. Data preprocessing 53
2.6 .Cleaning comments 53
3. Proposed Approach Steps 54
3.1 . Tokenization 54
3.2. Padding the sequence to a certain length 54
3.3 BERT Tokenization 54
Special token in Bert 55
3.4. Case conversion: 55
3.5.Word normalization 55
3.6. Punctuation and special character removal: 55
3.7. Transformation of Special Symbols: 55
3.8. Segmentation: 55
3.10. Creating sentence token attention masks: 56

4. 1. Word2Vec: 56
4.2. TFIDF 57
4. 3.Fasttext : 57
5.2. Deep learning : 57
5.3 Transfer learning 58
6. Evaluation Measures 61
Conclusion : 62
Chapter IV**: Test and Evaluation**
Introduction : 64
1.Environment and Development Tools : 64
2. Data Used for Training and Test : 67
2.1.Experimentation Steps : 68
2.1.1.For Pre-processing : 68
2.1.2. For Grouping : 70
2.1.3. For Trnscript letters : 71
2.1.4. For Pre-processing Bert models : 71
2.1.5. New Pre-processing : 72
3.Model Implementation 73
3.1.Dataset Preparation: 73
3.2.Models : 73
4. Test results for models 77
4.1.Models of machine learning : 77
4.2.Deep Learning Models : 81
4.3. For transfer learning : 84
4.4. Ensemble learning : 89
4.5. Comparison between (deep learning models and pretrained language models) And General (f1_score/ accuracy) comparison : 95
Bibliographie 99

List of figures

Figure.II.1. Machine learning algorithms 11

Figure.II.2. General Flow for Machine Learning. 12

Figure.II.3. Deep neural network. 14

Figure.II.4. A neuron can be visualized as this picture. 14

Figure.II.5 (a) Simplified representation of the previous artificial neuron. (b) More simplified representation. 14

Figure.II.6. The learning mechanism of a deep neural network 15

Figure.II.7. Illustration of deep neural network learning process.....16

Figure.II.8. RNN Architecture. 19

Figure.II.9. Basic Structure of LSTM. 21

Figure.II.10. The CNN baseline architecture is as follows.....22

Figure.II.11. Unrolled GRUs. 22

Figure.II.12. Architecture of the Transformer Model. 23

Figure.II.13. Multi-head scaled dot-product attention mechanism 25

Figure.II.14. The transformer's encoder. 26

Figure.II.15. The transformer's decoder. 26

Figure.II.16. Differences in pretraining model architectures. 27

Figure.II.17. Masked LM example.....28

Figure.II.18. Masked LM architecture of BERT.....29

Figure.II.19. Next sentence prediction example.....29

Figure.II.20 . BERT input representation.....29

Figure.II.21. The encoder-decoder model with additive attention mechanism.....31

Figure.II.22. The current word is in red and the size of the blue shade indicates the activation level.....33

Figure.III.1. General steps for constructing the annotated dataset.....48

Figure.III.2 An overview flowchart of the methodology.....49

Figure.III.3 The steps of data collection and preprocessing.....49

List of Figures

Figure.III.4. Information about Source.....50

Figure.III.5. Information about Topic.....51

Figure.III.6. Information about the collected dataset.....52

Figure.III.7. Information about the Topic.....52

Figure.III.8. Dl architecture 57

Figure.III.9. LSTM architecture.....58

Fgure.III.10 . BERT Model Representation. 59

Figure.III.11. BERT + Pooled Stacked Bi-LSTMs Model.....60

Figure.III.12. BERT + Pooled Stacked Bi-LSTMs Model.....60

Figure.III.13. Confusion Matrix.....61

Figure.IV.1. Exempl Dataset.....67

Figure.IV.2. New Pre-processing 71

Figure.IV.3. General steps for constructing the annotated dataset 1. 73

Figure.IV.4. General steps for constructing the annotated dataset 2. 74

Figure.IV.5. General steps for models. 75

Figure.IV.6. With TF-IDF embedding Accuracy. 79

Figure.IV.7. With Word2Vec embedding Accuracy. 79

Figure.IV.8. Without FastText Accuracy.....83

Figure.IV.9. with FastText Accuracy.....83

Figure.IV.10. BERT architecture. 83

Figure.IV.11. Distilbert-base-uncased/ Bert-base-multilingual-uncased/ Bert-base-arabic Accuracy.....85

Figure.Iv.12. The Bert with LSTM architecture. 86

Figure.IV.13. Bert with GRU architecture.....86

Figure.IV.14. Bert with GRU and LSTM.....86

Figure.IV.15. Accuracy Bert embedding with GRU and LSTM results.....88

Figure.IV.16. Ensemble learning Accuracy **1**.....92

Figure.IV.17. Ensemble learning Accuracy **2**.....93

List of Figures

Figure.IV.18. General f1_score comparison between deep learning models and pretrained language models.....95

Figure.IV.19. General accuracy comparison between deep learning models and pretrained language models.....95

List of Tables

List of tables

Table.I.1. The meaning of word (عين) as a noun. 8

Table.I.2. Affixes in arabic words. 8

Table.II.1. A summary table of several common attention mechanisms and their associated alignment score functions 32

Table.II.2. Summary of the current state of anti-social behaviour detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F) 40

Table.II.3. Summary of the current state of cyberbullying detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F) 40

Table.II.4. Summary of the current state of religion hate speech detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F).....41

Table.II.5. Summary of the Italian,Indonesia and German contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F) 43

Table.II.6. Summary of the Arabic contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F).....43

Table.II.7. Summary of the Algerian dialect contributions in hate speech detection and their respective results, in the metrics: Precision (P), Recall (R), F1-Score (F).....46

Table.III.1. Information about old data.....50

Table.III.2. Information about Source.....50

Table.III.3. Information about Topic.....50

Table.III.4. Information about Source.....51

Table.III.5. Information about Topic 52

Table.III.6. Information about the dataset 53

Table.III.7. Example from the dataset on hate speech.....53

Table.IV.1. Results of TF-IDF with ML. 76

Table.IV.2. Results With Word2Vec embedding.....78

Table.IV.3. Test results of datasets without FastTaxt 81

Table.IV.4. Test results of datasets with FastTaxt 82

Table.IV.5. fine-tune results. 84

Table.IV.6. Bert embedding with GRU and LSTM results. 87

List of Tables

Table.IV.7. results of ensemble models 1. 89

Table.IV.8. results of ensemble models 2. 92

Table.IIV.9. Comparison between (deep learning models and pretrained language models) And General (f1_score/ accuracy) comparison.....94

List of acronyms

AI: Artificial Intelligence.

AlBERT : A Lite BERT Version.

ANN: Artificial Neural Networks.

API: Application Programming Interface.

AraBERT: Pre-training BERT for Arabic Language Understanding.

AUC: Area Under the Curve

BERT: Bidirectional Encoder Representation from Transformers.

Bi-GRU: Bidirectional Gated Recurrent Unit.

Bi-LSTM: Bidirectional Long Short-Term Memory.

CBOW: Continuous Bag of Words.

CNN: Convolutional Neural Networks.

CPU: Central Processing Unit.

C&W: Collobert and Weston.

DistilBERT: Distilled Version of BERT.

DL: Deep Learning.

DNN: Deep Neural Network.

EL: Ensemble Learning.

ELMo: Embeddings from Language Models.

GLoMo: Graphs from LOw-level unit Modeling.

GloVe: Global Vectors for Word Representation.

GPT: Generative Pre-Training.

GPU: Graphics Processing Unit.

GRU: Gated Recurrent Unit.

LSTM: Long Short-Term Memory.

ML: Machine Learning.

MLM : Multi-level marketing.

NLP: Natural Language Processing.

NSP : Next Sentence Prediction

Introduction

Introduction

Introduction

In recent years, the use of the social networks has substantially increased in the Arab world. It has allowed more freedom for opinion expression. Moreover, organizations in the Arab region have embraced social media in their businesses at varying scales, assuming that it significantly affects business development. Due to the freedom of speech given to social media users, it has become relatively easy to propagate abusive or hate speech towards individuals, groups, or societies.

Manual analysis of hate speech on social media is impractical due to the sheer volume of data, as it is both expensive and time consuming. For this reason, it is important to detect and remove online hate speech by automatically processing user content.

Many means have emerged to detect the latter due to its importance, and it has attracted the attention of many researchers to detect hate speech. These methods include traditional methods, deep learning and transfer learning.

Traditional methods based on machine learning algorithms, such as support vector machines, and referring to lexicon based techniques, deep learning methods those have automated the features extraction using artificial neural networks like CNN and RNN. Finally, transfer learning methods which apply already learned knowledge in one domain to the new target domain taking into consideration the similarities between the two domains.

The Arabic language is spoken by 420 million people around the world, which makes it the sixth most spoken language in the international level. However, Arabic hate speech detection task was not tackled in early studies as for English and some other languages.

In Algeria, people speak Arabic, Darija and French, and use various multiple online social media platforms, including Reddit, YouTube, facebook , Twitter.. .The Arabic language and the Algerian dialect are very complex, which makes it difficult to detect hate speech.

As a result, early work on the Arabic hate speech language frequently included a pre-processing phase aimed at standardizing the dataset to make treatments easier and the study produce better results. General deep learning techniques and transfer learning techniques were used in this project.

Introduction

In this study, we focus on Algeria hate speech detection, with a dialectal Algeria dataset, including unstructured words, repeated letters, and sometimes…. We propose a pre-processing suitable to our data, and then, we propose an ensemble learning model, resulting from the combination of different deep learning architectures, based on BERT.

The study we made is reported in the next chapters as follow:

Chapter I: **Hate speech language Detection :**

The principles of offensive language, such as its terminologies, approaches applied by various researchers in early work, and applications, are presented in this first chapter. We talked about the negative effects of offensive language, particularly on children and women. We also discussed the need of recognizing offensive information and suggested strategies to combat it in some nations. Finally, we mentioned some early work on the subject.

Chapter II: **Deep Learning and transfer learning techniques** :

Chapter 2 shows and describes the various deep learning and transfer learning algorithms for detecting hate speech , as well as the techniques used by each approach. We introduce related work, which pertains to paper works on employing either deep learning or transfer learning models, with certain task-specific information.

Chapter III: **Proposed approach for Algerian hate speech detection**

The primary proposed techniques to dealing with the detection of hate speech in the Algerian dialect, as well as the interventions and contributions we've made at the data level and the planned approach steps, are presented in the third section. Machine learning, deep learning, and transfer learning classification are all examples of classification techniques.
Finally, we talked about which indicators we used to assess our model's success.

Chapter IV**: Test and Evaluation**

In this chapter, we'll go over the environment, the dataset, and the Experimentation Steps, as well as the many modules we use to trainand test our models. The processes we use to construct our dataset are then shown, followed by test results for each model.

Chapter I

Hate speech language Detection

Introduction

Violent language is the root of all difficulties that people confront on a daily basis. Because violence and verbal violence infiltrate every aspect of human life. Language scientists are currently conducting extensive research on violent language in the hopes of one day being able to govern and legitimize human writing.

In this chapter, we'll go through the terminology for verbal violence, as well as the effects and potential solutions.

1. Definitions and Terminology

1.1. Violence

Violence is the use of physical force so as to injure, abuse, damage, or destroy"The intentional use of physical force or power, threatened or actual, against oneself, another person, or against a group or community, that either results in or has a high likelihood of resulting in injury, death, psychological harm, maldevelopment or deprivation." (Krug E, Dahlberg L, Mercy J., 2002)

1.2. Offensive Language

Offensive language is the offence of using language in a way which could cause offence to a reasonable person in, near, or within hearing or view of a public place or school. The categories are cursing, profanity and blasphemy, epithets, obscenity and insults (Jay, 1992)

Offensive languages are used to express the strong emotion as anger, dissatisfaction, and shocked feeling. They generally involve something society deems sacred (religion, family) or something considered taboo or prohibited or are used to insult someone personally (Sinambela, 2010)

1.3. Hate speech

Hate speech is more than just harsh words. It can be any form of expression intended to vilify, humiliate, or incite hatred against a group or class of people. It can occur offline or online or both. It can be communicated using words, symbols, images, memes, emojis and video.

According to the Cambridge Dictionary, hate speech is defined as "public communication that expresses hatred or supports violence against a person or group on the basis of anything like race, religion, sex, or sexual orientation" According to the definition, "usually understood to entail expressions of hatred or disparagement of a person or a community on the basis of a group feature such as race, color, national origin, sex, handicap, religion, or sexual choice" (Nockleby, J. T., 2000)The legal definition of hate speech vary from country to country.

1.4 hate speech en internet

Due to many psychological characteristics, hate speech is more common on the Internet than in real life (Brodnig, I, 2016)Individual users' rude and nasty remarks to socially and politically problematic themes in social media and online comments are common. This harsh and abusive language, if not moderated properly, can swiftly escalate arguments or cause participants to withdraw (Springer, N., Engelmann, I., and Pfaffinger C.)

1.5. Bullying

Bullying is defined as the abuse, aggressive dominance, or intimidation of another person by the use of force, compulsion, harsh taunting, or threat. The pattern of conduct is repetitive and habitual. One of the most important prerequisites is the impression of an imbalance of physical or social power (by the bully or others). Bullying is distinguished from disagreement by this imbalance (Juvonen, J., & Graham, S. , 2014)"Bullying in Schools: Bullies' Power and Victims' Tragedy" International, harmful, and repetitive behavior by one or more people.Over time, the conduct is repeated, or has the potential to be repeated.

Actions such as issuing threats, spreading stories, physically or verbally assaulting someone, and purposefully excluding someone from a group.

2. Detrimental Effects of offensiveness

2.1. Violence against women

Violence against women is now well recognised as a public health problem and human rights violation of worldwide significance. It is an important risk factor for women's ill health, with far reaching consequences for both their physical and mental health. This glossary aims to describe various forms of interpersonal violence that are directed towards women and girls.

The term violence against women encompasses a multitude of abuses directed at women and girls over the life span. The UN Declaration on the Elimination of Violence against Women (defines violence against women as: ''….any act of gender-based violence that results in, or is likely to result in physical, sexual or psychological harm or suffering to women, including threats of such acts, coercion or arbitrary deprivation of liberty, whether occurring in public or in private life'' (Campbell, J. C. Health consequences of intimate partner violence, 2002)

2.2. Violence against children

Verbal abuse can lower a child's self-esteem, harm his ability to trust and form relationships, and erode his academic and social abilities. The latest research suggests that verbal abuse of children can be just as emotionally harmful as physical and sexual abuse, placing them at risk for depression and anxiety (Youth Violence Prevention Resource Center, 2005) (US Department of Justice. National Crime Victimization Survey, 2006) (Harvard Mental Health Letter, 2007).

3. Proposed Automated Offensiveness Detection Solutions

3.1. Inherent Need of Automated Tool for Offensiveness Detection

Living in harmony is one of the most significant aspects of leading a normal and healthy life. Individuals can achieve this degree of peace by removing themselves from all forms of abuse and violence, which would improve their capacity to be productive members of society. The rate of violence has been increasing all over the world in recent years, and this is a sign of danger that societies should be aware of to take the appropriate measures to prevent or minimize it. Physical, sexual, and psychological abuse are all forms of acts that fall under the definition of violence.

Because of the development of technology and the rising number of internet users around the world, they are becoming a significant environment for studying phenomena related to abuse, especially when it comes to using books and papers and articles. In this light, the current study focuses on the increased number of violence cases in the Kingdom of Saudi Arabia (KSA), where the Ministry of Labor and Social Development received over 11,000 reports in a single year (Alshehri. M)

3.2. Real World Systems for Offensiveness Detection

This type of research is valuable to various organizations such as Human Rights Organizations (HRO). Arabic is the official language in around 17 countries with more than 315 million people around the world. All of these countries are Diglossia societies where both the standard form of the language, Modern Standard Arabic (MSA), and the regional dialects (DA) are used (Elfardy & Diab, 2013). MSA is used in official environments, while Arabic speakers speak DA as their first language. There is no standard orthography for DA, so it is divided into many classes among these countries.

The task of detecting violence in Arabic books is not easy. owing to the fact that violent Arabic terms are not necessarily indicative of a violent background The word "killing," for example, has a violent connotation but can also be used in a non-violent sense.Examples:

"The memory and the pain are sisters, and you can't kill the pain without killing the memory," says the narrator.

"You can ruin the flowers, but you can't stop spring from coming."

They used text mining, which is a hot subject in terms of the information it uncovers. Information retrieval, categorization, extraction, textual analysis, and visualization are all part of text mining, which is characterized as a technique for extracting useful information from text. The greatest difficulty in text classification, after information extraction, is the vast number of features that must be considered (Hotho et al., 2005). Featurereducing techniques are used to select the appropriate subset to solve this problem. Feature reduction is a key move that can have a positive or negative effect on the classifier's accuracy depending on the approaches used (Liu & Motoda, 2011).

This study investigated different feature-reduction methods and their effects on accuracy. Moreover, it used machine learning techniques, which can be divided into two categories: supervised and unsupervised. The data are labeled in the supervised approach, and the algorithm works to predict the output; in the unsupervised approach, the data are not labeled, and the algorithm works to construct structures to interpret the data (Maglogiannis et al., 2007).

4. Proposed solution hate speech detection :

4.1 The need of automatic hate speech detection:

As seen by increased media coverage and government attention to the subject in recent years, hate speech has become a more common crime. There are a variety of reasons to focus on automatic hate speech detection, as explained below:

- **Directives of the European Union Commission:** In recent years, the European Union Commission has launched a variety of programs targeted at curbing hate speech. To fight hate speech, a slew of new programs are springing up. The European Union

Commission has pressed Facebook, YouTube, Twitter, and Microsoft to sign an EU hate speech code, which demands that the majority of genuine notifications be checked in less than 24 hours in order to eliminate illegal hate speech.[1]

- **Automatic techniques not available:** Automated techniques tend to programmatically label text as hate speech, making its identification quicker and simpler for those in charge of public safety. Although some research has been done on the automatic detection of hate speech, the resources available are limited.

- **Lack of data about hate speech:** There is a general scarcity of systematic monitoring, documentation, and data collection of hate and violence.

- Hate speech reporting and exclusion might be a source of worry for certain companies and media outlets. Online media publishers and platforms in general, for example, must attract advertising and so cannot risk being classified as hate speech platforms .

- **Quality of service:** Companies that operate on social media offer a service. They make it easier for consumers to communicate with one another. They benefit from this program and, as a result, assume public responsibility for the information exchanged. In this situation, providing quality support in regards to hate speech entails taking measures to stop online hate and delete hate speech in a timely manner. Both can be calculated and compared to a reference point. Both can be measured and contrasted to a norm.

4.2 Hate speech detection using automated methods:

Most social media sites have created user rules prohibiting hate speech; however, enforcing these rules involves a lot of manual effort to go through each report. Facebook, for example, has expanded the number of content moderators. Automatic technologies and procedures might speed up the assessment process or reallocate human resources to positions that demand in-depth human scrutiny. In this part, we look at how to detect hate speech from text using automated methods.

- **Approaches that use keywords:**A keyword-based technique is a simple strategy for identifying hate speech. Text that contains potentially hostile terms is recognized using an ontology or dictionary. Hatebase[2], for example, maintains a collection of pejorative names for a variety of groups in 95 languages. Because language develops with time, such well-maintained materials are quite helpful. However, merely uttering a racial slur isn't always enough to qualify as hate speech.

Metadata sources: Additional data from social media can aid in a better understanding of the features of the postings and, as a result, a more accurate identification method. Demographics of the posting user, location, timing, and even social involvement on the site can all provide further insight into the post at various levels of detail.

[1] Alex Hern. 2016. Facebook, YouTube, Twitter, and Microsoft sign EU hate speech code. https://www.theguardian.com/technology/2016/may/31/facebook-youtube-twitter-microsoft-eu-hate-speech-code.

[2] https://hatebase.org/.

External researchers seldom get access to this data since releasing data including sensitive user information creates privacy concerns. External researchers may only have access to a portion of or none of the user data. As a result, kids may answer the erroneous problem or learn based on incorrect data understanding. For example, depending on incidental dataset features, a system trained on these data can naturally lean towards recognizing material by specific persons or groups as hate speech.

- **Machine learning classifiers :**Machine learning models use samples of tagged text to create a classifier that can detect hate speech based on content reviewer annotations. Various models have been presented in the past and have proven to be successful.

To categorize information as hate speech, the suggested methods used several feature engineering approaches and machine learning algorithms. Despite this enormous amount of effort, comparing the efficacy of several techniques to categorize hate speech material remains challenging. To the best of our knowledge, there are no studies that compare different feature engineering approaches and machine learning algorithms.

5. Arabic language

With more than 315 million people speaking Arabic around the world, it is one of the most widely spoken languages today. Furthermore, Arabic is the official language of 17 countries in the Arab world (Elfardy,H., & Diab.M, 2013).

Countries where Arabic is the primary spoken language: Algeria, Bahrain, Djibouti, Egypt, Eritrea, Iran, Iraq, Palestinian , Jordan, Kuwait, Lebanon, Libya, Mauritania, Morocco, Oman, Palestinian West Bank & Gaza, Qatar, Saudi Arabia, Somalia, Sudan, Syria, Tanzania, Tunisia, United Arab Emirates, and Yemen. (Classical Arabic is read everywhere that Islam ispracticed.).

There are 3 functional types of Arabic language:

1. Classical Arabic is the language of the Qur'an, and is used primarily for reading and reciting Islamic holy text
2. Spoken Arabic has 4 regional dialects, but over 30 actual varieties of colloquial (spoken) Arabic around the world
3. Modern Standard Arabic is used in the news and on TV the "common language" used byspeakers of different dialects.

Arabic consists of 28 letters, which are :

(أ-> aleph),(ب -> bet),(ت -> taw),(ث -> ta) ,(ج -> gimel), (ح-> heth),(خ-> ha), (د->dalet),(ذ ->dal),(ز->zayin),(ر->resh),(س->sin),(ش->shin),(ص->tsad) ,(ض->daad),(ط->teth),(ظ->za),(ع->ayin),(غ->ghain),(ف->pe),(ق->Qoph),(ك->kaph),(ل->lamedh),(م->mim),(ن->nun),(ه->he),(و->waw),(ي->yaa), additional are there a vowels letters و ا ي. Arabic.

Most letters change form and look different when they appear at the beginning, in the middle, or at the end of a word .Exemple :

The letter (ه)is written in one way if it comes at the beginning of a word, such as (هرة)poon, (ـهـ)another way if it comes in the middle of the word, such as (جمهور)public, and yet another

way if it comes at the end of the word (هـ)such as (دمه) hiss blood (Saad, M. K., & Wesam Ashour, 2010).

There are two genders in Arabic words: masculine (مذكر) and feminine (مؤنث); three numbers: single(مفرد مثل: قلم) , dual (مثنى مثل: قلمان), and plural(جمع مثل :أقلام); and three grammatical cases: nominative(ُ الرفع) , accusative(َ النصب) , and genitive (ِ الجر). Words are classified into three main parts of speech, nouns (الاسماء) including verbs (الافعال), adverbs (ظروف) ,adjectives (صفات) and particles (الادوات) .

Word meanings: Because a single word can have several meanings in different circumstances, it is possible to determine the various meanings connected with it. Table 1 depicts the Arabic noun (عين), which has three meanings.

Table.I.1. The meaning of word (عين) as a noun.

Word meanings	**Sentence**
Eyes	لون عينيه ازرق
spelling letter	العين حرف هجاء
Envy	العين حق
Ain	عين الذهب منطقة في ولاية المدية

All Arabic words have a root, which may be defined as the most significant component of any word that cannot be eliminated without causing the term to lose its meaning. More than 11,000 roots exist in Arabic (Al-Zabidi, 1965). Prefixes, suffixes, and middle affixes are all examples of affixes (infixes). See the word example in Table III (Saad M. K., Ashour W., 2010)for a more detailed explanation (Farghaly & Shaalan, 2010). All of these features of Arabic illustrate the challenges that researchers encounter while mining data written in the language.

Table.I.2. Affixes in arabic words

Suffix	**Infix**	**Prefix**	**Word**
ية	/	/	قمرية
/	ي	ت	تسريع
ية	ا	است	استخراجية
/	و	/	علوم

Problems with Arabic Dialect:

- Language that isn't organized.
- Errors in orthography.
- informal language used in social media.
- Arabic does not have capital letters, which would otherwise be used to mark features.
- lack of resources needed.
- no grammar in dialects.
- the different local informal dialect for each city.

6. Challenges in Hate Speech Detection:

Hate speech is a complex phenomena that might be difficult to spot. The following impediments and challenges were mentioned by the writers of the surveyed articles:

• When it comes to identifying hate speech, humans have a low degree of agreement, meaning that machines would have a harder job.
• This purpose necessitates cultural and social structural understanding.
• When social phenomena and lexicon change, it's hard to keep track of both ethnic and minority insults.
• Language evolves quickly, especially among young people who speak on a regular basis.
• Language develops quickly, especially among young people who use social media to interact often.
• Inflammatory rhetoric can be quite fluent and grammatically correct, break sentence boundaries, and contain sarcasm, despite the fact that hate speech is banned.
• Finally, detecting hate speech takes more than merely searching for keywords.

7. Fighting hate speech in Algerian:

ALGIERS- President of the Republic Abdelmadjid Tebboune has instructed Prime Minister Abdelaziz Djerad to develop a bill criminalizing all forms of racism, regionalism and hate speech in the country, the presidency of the Republic said in a statement in 13/01/2020.

Conclusion

The hate speech detection problem was examined in this chapter. We began by outlining basic definitions of hate speech and the various forms it might take. Second, we spoke about how inflammatory language has a harmful influence on children and women in particular. After that, we demonstrated how abusive behavior has a negative impact on users' physical and emotional health. We also looked into other relevant research as well as the approaches for detecting offensive language and hate speech. We also spoke about the existing automated approaches to dealing with hate speech in online environments. The obstacles that researchers experienced in this field were then discussed, We discussed the Arabic language and the difficulties we encounter with it, as well as Algeria's strategy to combating hate speech.

Chapter II

Deep learning and transfer learning techniques

Part 1: Introduction to Deep Learning and Transfer learning

Introduction :

With the advancement of web technology,storage capabilities, and Internet exchanges. In terms of data volume and user numbers, social networks are seeing a worldwide expansion. The everyday usage of social media platforms such as Facebook, Twitter, and Instagram has altered the perception of web 2.0, giving it a new dimension as well as new obstacles. With such a massive amount of data, social networks have lately become a particularly appealing subject of research for many societal players, including news organizations, businesses, computer and information science academics, psychologists, and sociologists. These study topics primarily address market research, advertising campaign monitoring, trend analysis, human, societal, and individual behavior analysis, illness detection, and the identification of prominent persons.

1. Machine learning :

Machine Learning is an artificial intelligence approach. It is a subset of artificial intelligence (AI) in the area of computer science that uses statistical approaches to allow computers to "learn" (i.e., steadily improve performance on a given job) from data without having to be programmed. (Alpaydin, 2020)

1.1. Supervised learning

Systems that are exposed to vast volumes of labeled data are used. To master a mission, some systems may require exposure to millions of examples (Alpaydin, 2020).

Supervised learning is the most popular paradigm for machine learning. It is the easiest to understand and the simplest to implement. Given data in the form of examples with labels, we can feed a learning algorithm these example-label pairs one by one, allowing the algorithm to predict the label for each example, and giving it feedback as to whether it predicted the right answer or not. Over time, the algorithm will learn to approximate the exact nature of the relationship between examples and their labels.

1.2. Unsupervised learning

Refers to programs that attempt to spot trends in data to divide them into categories (Alpaydin, 2020).

Unsupervised learning is very much the opposite of supervised learning. It features no labels. Instead, our algorithm would be fed a lot of data and given the tools to understand the properties of the data. From there, it can learn to group, cluster, and/or organize the data in a way such that a human (or other intelligent algorithm) can come in and make sense of the newly organized data.

1.3. Reinforcement (Semi-supervised) learning

Combines supervised and unsupervised learning to train systems with small amounts of labeled and large amounts of unlabeled data. The labeled data is used to partially train a machine-learning model, which is then used to mark the unlabeled data in a method known as pseudo-labeling (Alpaydin, 2020).

1.4. Machine learning algorithms

To tackle data challenges, Machine Learning employs many algorithms. Data scientists like to point out that there is no single optimum algorithm for every problem. The type of method used is determined by the type of issue you're trying to answer, the amount of variables involved, the best model for the job, and so on. Here are some of the most regularly used machine learning algorithms (ML)

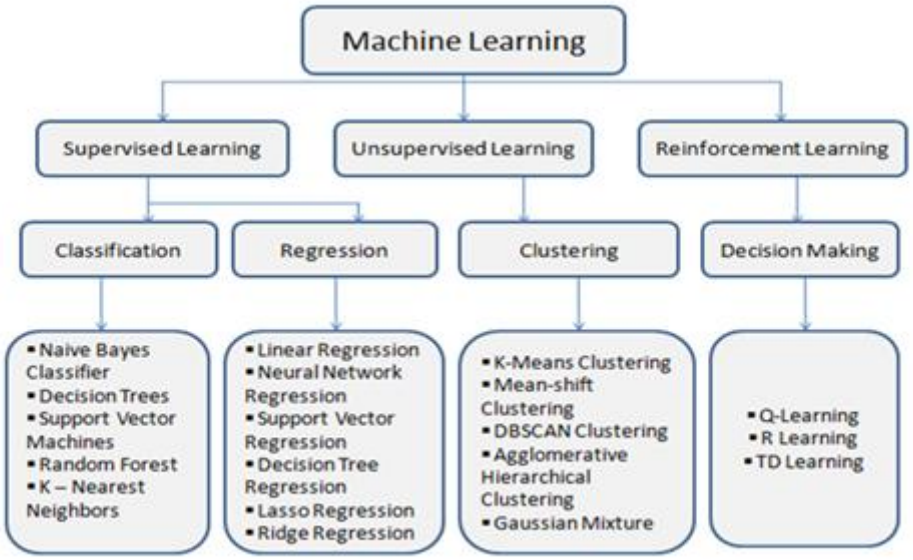


Figure.II.1. Machine learning algorithms.[3]

Machine Learning algorithm is an evolution of the regular algorithm. It makes your programs "smarter", by allowing them to automatically learn from the data you provide. The algorithm is mainly divided into:

1. Training Phase
2. Testing phase

Training Phase You take a randomly selected specimen of apples from the market (training data), make a table of all the physical characteristics of each apple, like color, size, shape, grown in which part of the country, sold by which vendor, etc (features), along with the sweetness, juiciness, ripeness of that apple (output variables). You feed this data to the machine learning

[3] https://www.analyticsvidhya.com/blog/2021/03/everything-you-need-to-know-about-machine-learning/

algorithm (classification/regression), and it learns a model of the correlation between an average apple's physical characteristics, and its quality.

Testing Phase Next time when you go shopping, you will measure the characteristics of the apples which you are purchasing(test data)and feed it to the Machine Learning algorithm. It will use the model which was computed earlier to predict if the apples are sweet, ripe and/or juicy. The algorithm may internally use the rules, similar to the one you manually wrote earlier (for eg, a decision tree). Finally, you can now shop for apples with great confidence, without worrying about the details of how to choose the best apples.

1.5. Feature Engineering in Machine Learning

Most machine learning examples assume the features (**the variables describing the input to be modeled)** are given or easy to determine; however, this may not be the case in the real world. Researching and including features that enhance are essential in emerging and advanced models; such as, those for precision medicine, autonomous vehicles, drones, and 5G / Internet of Things (IoT). The new and the advanced may be the norm as over 31 billion devices are expected to be internet connected by the end of 2020 and over 75 billion by 2025 (Horwitz, 2019). Perhaps feature engineering will be the evolving branch of data science given the onslaught of data and intelligent processors.

Features, in the context of machine learning, are the inputs and feature engineering is commonly characterized as:

- Creating new input features from your existing ones (Science, 2020)
- The process of using domain knowledge of the data to create features that make machine learning algorithms work (Shekhar, 2018)

Coming up with features is difficult, time-consuming, and requires expert knowledge. 'Applied machine learning' is basically feature engineering (Andrew,Ng., 2013)

Text mining tasks such as document classification and sentiment analysis use feature engineering extensively. The feature engineering method is required to construct features involving word/frequency frequencies because individual pieces of raw text are normally used as input data (Guozhu Dong,Huan Liu ., 2018).

Figure provides a simplified view of how feature selection and engineering fit into the overall flow of machine learning.

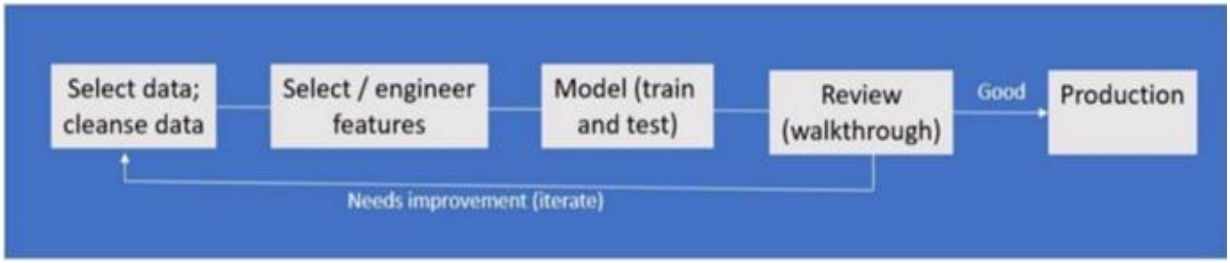


Figure.II.2. General Flow for Machine Learning.

2. Deep Learning (DL)

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before .In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers .[4]

2.1 The importance of deep learning

in a word, accuracy. Deep learning achieves recognition accuracy at higher levels than ever before. This helps consumer electronics meet user expectations, and it is crucial for safety-critical applications like driverless cars. Recent advances in deep learning have have improved to the point where deep le improved to the point where deep learning outperforms humans in some tasks like classifying objects in images

While deep learning was first theorized in the 1980s, there are two main reasons ithas only recently become useful:

1. Deep learning requires large amounts of **labeled data**. For example, driverless car development requires millions of images and thousands of hours of video.
2. Deep learning requires substantial **computing power**. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

2.2. How Deep Learning Works

Most deep learning methods use **neural network** architectures, which is why deep learning models are often referred to as **deep neural networks** .The term "deep" usually refers to the number of hidden layers in the neural network. Traditional neural networks only contain 2-3 hidden layers, while deep networks can have as many as 150.Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction.

[4] https://www.mathworks.com/discovery/deep-learning.html

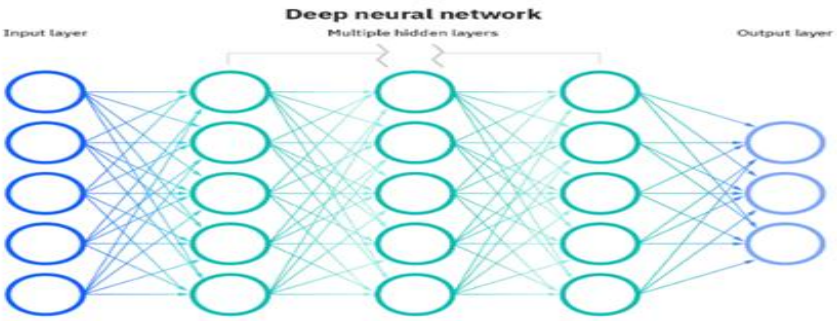


Figure.II.3. Deep neural network [5]

2.3. Learning Process of a Deep Neural Networks

A neural network is made up of neurons connected to each other; at the same time, each connection of our neural network is associated with a weight that dictates the importance of this relationship in the neuron when multiplied by the input value. **Figure.II.4** below shows the neuron vision:

A neural network is made up of **neurons** connected to each other; at the same time, each connection of our neural network is associated with a weight that dictates the importance of this relationship in the neuron when multiplied by the input value.

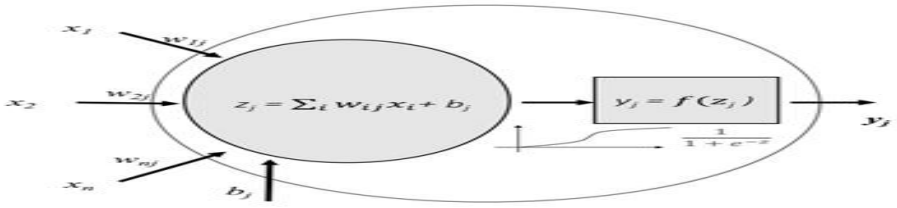


Figure.II.4. A neuron can be visualized as this picture.

Each neuron has an **activation function** that defines the output of the neuron. The activation function is used to introduce non-linearity in the modeling capabilities of the network. We have several options for activation functions that we will present in this post

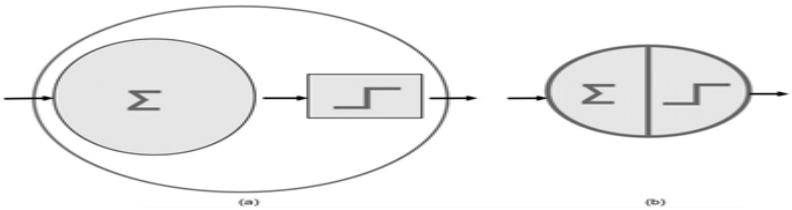


Figure.II.5. (a) Simplified representation of the previous artificial neuron. (b) More simplified representation[6]

[5] https://www.ibm.com/cloud/learn/neural-networks

The most genuine part of Deep Learning is training our neural network, which is learning the values of our parameters (wij weights and bj biases). We can see this learning phase in a neural network as an iterative process of "going and returning" by the layers of neurons. The "moving" is information that is propagated forward, and the "return" is information that is propagated backward.

The first step of forward propagation occurs when the network is exposed to training data, which is then used to quantify predictions (labels) across the entire neural network. That is, moving the input data through the network in such a way that all of the neurons convert the information they obtain from the previous layer's neurons before sending it to the next layer's neurons. When the data has crossed all the layers, and all its neurons have made their calculations, the final layer will be reached with a result of label prediction for those input examples.

Next, you use the loss function to estimate the loss (or error) and to compare and measure how good/bad the outcome of our prediction is about the correct outcome. Ideally, it should be without difference between the estimated and expected value. Therefore, during model training, the correlation weights between neurons are gradually adjusted until good predictions are obtained

Once the loss has been calculated, this information is propagated backward. Hence, its name: back propagation. Starting from the output layer, that loss information propagates to all the neurons in the hidden layer that contribute directly to the output. However, the neurons of the hidden layer only receive a fraction of the total signal of the loss, based on the relative contribution that each neuron has contributed to the original output. This process is repeated, layer by layer, until all the neurons in the network have received a loss signal that describes their relative contribution to the total loss.

We can visually summarize what we've explained in the Figure 6. below:

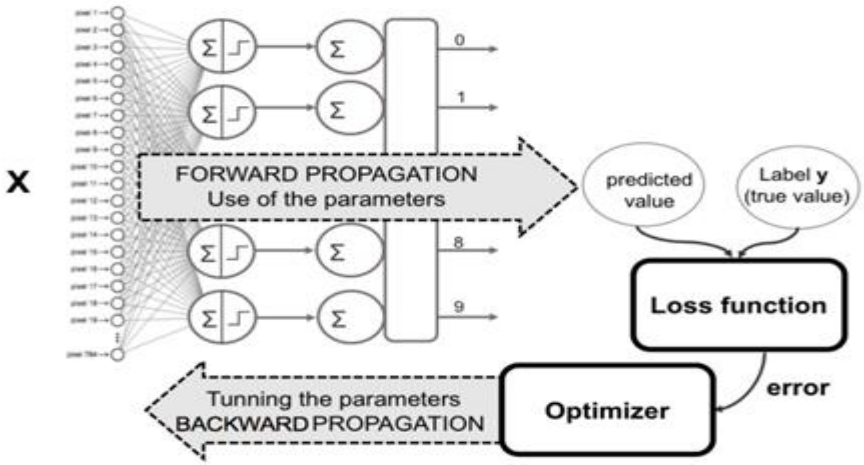


Figure.II.6. The learning mechanism of a deep neural network

[6] https://torres.ai

Now that we have spread this information back, we can adjust the weights of connections between neurons. What we are doing is making the loss as close as possible to zero the next time we go back to using the network for a prediction. For this, we will use a technique called **gradient descent**. This technique changes the weights in small increments with the help of the calculation of the derivative (or gradient) of the loss function, which allows us to see in which direction "to descend" towards the global minimum; this is done in general in batches of data in the successive iterations (epochs) of all the dataset that we pass to the network in each iteration.
To recap, the learning algorithm consists of:

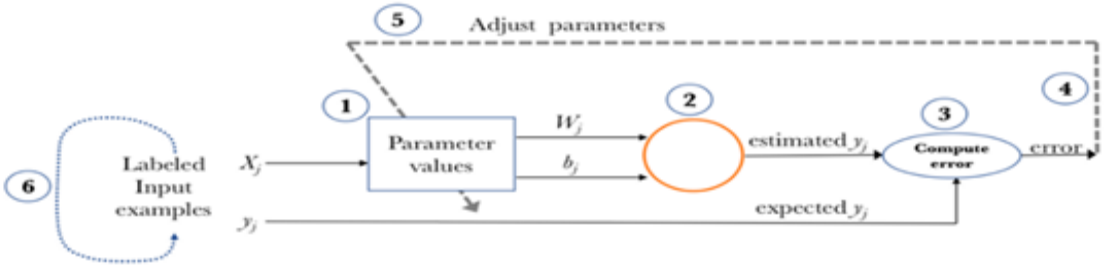


Figure.II.7. Illustration of deep neural network learning process. (TORRES.AI, 2018)**.**

1.Start with values (often random) for the network parameters (*wij* weights and *bj* biases).
2.Take a set of examples of input data and pass them through the network to obtain their prediction.
3.Compare these predictions obtained with the values of expected labels and calculate the loss with them.
4.Perform the back propagation in order to propagate this loss to each and every one of the parameters that make up the model of the neural network.
5.Use this propagated information to update the parameters of the neural network with the gradient descent in a way that the total loss is reduced and a better model is obtained.
Continue iterating in the previous steps until we consider that we have a good model

2.3.1. Activation function

Remember that we use the activation functions to propagate the output of a neuron forward. This output is received by the neurons of the next layer to which this neuron is connected (up to the output layer included). As we have said, the activation function serves to introduce non-linearity in the modeling capabilities of the network. Below we will list the most used nowadays; all of them can be used in a layer of Keras.

- Linear : The linear activation function is basically the identity function in which, in practical terms, it means that the signal does not change.
- Sigmoid The sigmoid function has already been introduced in **a previous post**. Its interest lies in the fact that it allows a reduction in extreme or atypical values in valid data without

eliminating them: it converts independent variables of almost infinite range into simple probabilities between 0 and 1. Most of its output will be very close to the extremes of 0 or 1.

- Tanh Without going into detail, we can summarize that the tanh represents the relationship between the hyperbolic sine and the hyperbolic cosine: tanh(x)=sinh(x)/cosh(x). Unlike the sigmoid function, the normalized range of tanh is between -1 and 1, which is the input that goes well with some neural networks. The advantage of tanh is that negative numbers can be dealt with more easily.
- Softmax The softmax activation function was also presented in **a previous post** to generalize the logistic regression, insofar as instead of classifying in binary it can contain multiple decision limits. As we have seen, the softmax activation function will often be found in the output layer of a neural network and return the probability distribution over mutually exclusive output classes.
- ReLU The activation function rectified linear unit (ReLU) is a very interesting transformation that activates a single node if the input is above a certain threshold. The default and more usual behavior is that, as long as the input has a value below zero, the output will be zero but, when the input rises above, the output is a linear relationship with the input variable of the form $f(x)=x$. The ReLU activation function has proven to work in many different situations and is currently widely used.

2.3.2. Backpropagation components

Back propagation can be thought of as a way of adjusting the neural network's parameters (weights and biases) in the right direction. It begins by calculating the loss term and then adjusts the parameters of the neural network in reverse order using an optimization algorithm that takes this measured loss into account.

2.4.3. Loss function

a loss function is one of the parameters used to determine how similar a neural network is to its ideal weight during the training phase

- **MSE(L2 error)** measures the average squared difference between the actual and predicted values by the model. The output is a single number associated with a set of values. Our aim is to reduce MSE to improve the accuracy of the model.
- **Binary Cross entropy (BCE):** For binary classification tasks, BCE loss is used. When using the BCE loss function, you only need one output node to divide the data into two categories. The output value should be passed through a sigmoid activation function, with the output range being (0–1) (Creswell, 2017)
- **Hinge loss** This type of loss is used when the target variable has 1 or -1 as class labels. It penalizes the model when there is a difference in the sign between the actual and predicted class values.
- **Categorical Cross entropy (CCE) :**These are similar to binary classification cross-entropy, used for multi-class classification problems, If you're using the CCE loss function, the

output nodes must be the same number as the groups. And the final layer output should be softmax enabled so that each node outputs a probability value between 0 and 1.

- **Kullback Leibler Divergence Loss** calculates how much a given distribution is away from the true distribution. These are used to carry out complex operations like autoencoder where there is a need to learn the dense feature representation.
- **Sparse Categorical Cross entropy (SCCE):** This loss function is almost similar to CCE except for one change. When we are using SCCE loss function, you do not need to one hot encode the target vector. If the target image is a cat, simply enter 0; otherwise, enter 1. Basically, you only pass the index of the class you want to use.
- **Focal Loss:** was introduced by (Lin et al, 2017)from Facebook the one-stage object detection scenario in which there is an extreme imbalance between foreground and background classes during training in dense object detection. It is a cross-entropy loss that weighs the contribution of each sample to the loss based in the classification error.

2.3.4. Optimizer

The learning technique is a global optimization problem in which the parameters (weights and biases) must be modified. In most cases, these parameters cannot be solved analytically, but they can be addressed effectively with optimizing algorithms in general.

2.3.5. Gradient descent

Many optimizers use gradient descent as their foundation, and it is one of the most widely used optimization algorithms in machine learning and deep learning. When modifying the parameters, it uses the loss function's first derivative (gradient). The frequency at which parameters are updated is adjustable.

Updating the parameters, gradient descent uses the first derivative (gradient) of the loss function. The method entails chaining the derivatives of each hidden layer's loss from the derivatives of its upper layer's loss, as well as integrating its activation function into the calculation (which is why the activation functions must be derivable). Once all of the neurons have the value of the gradient of the loss function that corresponds to them in each iteration, the values of the parameters are changed in the opposite direction. The gradient always points in the direction of the loss function's increasing value (TORRES.AI, 2018).

2.4. Deep Learning Performance

Computer and deep learning models' output rises in direct proportion to the amount of training data available. Since it is suggested to train such models on more data in order for them to generalize better to new data, advancements in hardware and the exponential increase in the amount of training data made available are the key reasons for deep learning's widespread effects

2.5. Popular Deep Learning Algorithms

2.5.1. Deep Neural Network

Deep Neural Networks (DNNs) are typically Feed Forward Networks (FFNNs) in which data flows from the input layer to the output layer without going backward[3] and the links between the layers are one way which is in the forward direction and they never touch a node again.

2.5.2. Recurrent Neural Networks (RNNs)

A Recurrent Neural Network (RNN) addresses this issue which is a FFNN with a time twist. This neural network isn't stateless, has connections between passes and connections through time. They are a class of artificial neural network where connections between nodes form a directed graph along a sequence like features links from a layer to previous layers, allowing information to flow back into the previous parts of the network thus each model in the layers depends on past events, allowing information to persist.

In this way, **RNNs** can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. But they not only work on the information you feed but also on the related information from the past which means whatever you feed and train the network matters, like feeding it 'chicken' then 'egg' may give different output in comparison to 'egg' then 'chicken'. **RNNs** also have problems like vanishing (or exploding) gradient/long-term dependency problem where information rapidly gets lost over time. Actually, it's the weight which gets lost when it reaches a value of 0 or 1 000 000, not the neuron. But in this case, the previous state won't be very informative as it's the weight which stores the information from the past.

Figure 17 **.** shows a simple architecture of RNN is working on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer. where Xt is input tokens, U is weighted for the connections between the input layer and hidden layers, W is the weight for connections between the hidden layer and output layer, ht is activations of hidden layer neurons and Ot is output.

First, it takes the Xt-1 from the sequence of input, and then it outputs Ot-1 which together with Xt is the input for the next step. So, the Ot-1 and Xt are the input for the next step. Similarly, Ot from the next is the input with Xt+1 for the next step and so on. This way, it keeps remembering the context while training.Figure 8 below shows the RNN architecture:

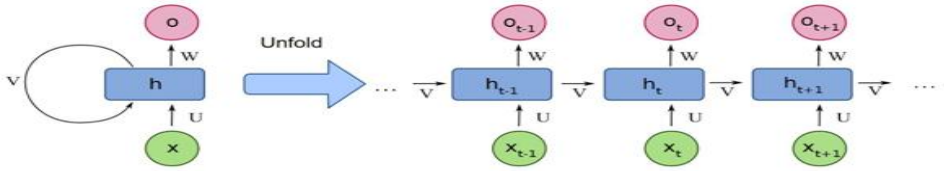


Figure.II.8. RNN Architecture.[7]

[7] https://miro.medium.com/max/4136/1*SKGAqkVVzT6co-sZ29ze-g.png

The feed-forward neural network had a few flaws, which led to the development of recurrent neural networks:

1. Cannot handle sequential data.
2. Considers only the current input.
3. Cannot memorize previous inputs.

The Recurrent Neural Network is the answer to these problems (RNN). An RNN can deal with sequential data by accepting both current and previously obtained inputs. Because of their internal memory, RNNs can memorize previous inputs.

Modeling time-dependent and sequential data problems, such as stock market prediction, machine translation, and text generation, with recurrent neural networks is possible. However, because of the gradient problem, recurrent Neural Networks are difficult to train.

The problem of vanishing gradients affects RNNs. Gradients transport information to the RNN, and when the gradient is too small, parameter changes become meaningless. Long data sequences are difficult to understand because of this.

2.5.3. Long Short-Term Memory (LSTM) Network

breakthroughs like **Long Short Term Memory (LSTM)** don't have this problem! **LSTMs** are a special kind of **RNN**, capable of learning long-term dependencies which make **RNN** smart at remembering things that have happened in the past and finding patterns across time to make its next guesses make sense. **LSTMs** broke records for improved Machine Translation, Language Modeling and Multilingual Language Processing.

In the area of deep learning, long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture (Hochreiter& Schmidhuber , 1997)LSTM has feedback connections, unlike normal feed forward neural networks. It can not only process only single data points (such as images), but also entire data sequences (such as speech or video). For example, LSTM can be used for tasks like unsegmented, linked handwriting recognition (Graves, Liwicki, Fernandez, Bertolami, Bunke, & Schmidhuber, 2009) speech recognition (Sak et al,, 2017)and anomaly detection in network traffic.

Since there may be lags of uncertain length between important events in a time series, LSTM networks are well-suited to classifying, processing, and making predictions based on time series data. LSTMs were created to solve the problem of vanishing gradients that can occur while training conventional RNNs .In many applications, LSTM has an advantage over RNNs, and other sequence learning methods due to its relative insensitivity to gap length.

An LSTM has three gates and a memory cell state from which information can flow from one module to another with either some linear interactions or unchanged. The gates are

means to remove or add information to the cell states, and they consist of a multiplication operation and a sigmoid layer. The three gates in an LSTM module are input, forget, and output gates:

Forget Gate :

Controls what information to throw away from memory.

Decides how much of the past you should remember.

Update/Input Gate :

Controls what new information is added to cell state from current input.

Decides how much of this unit is added to the current state

Output Gate :

Conditionally decides what to output from the memory.

Decides which part of the current cell makes it to the output. (Dessí et al ., 2020)

Figure 9 shows the architecture of a LSTM cell:

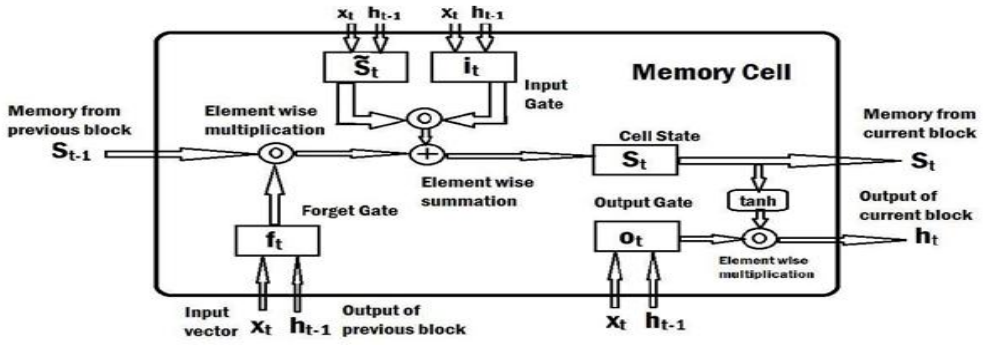


Figure.II.9. Basic Structure of LSTM.

2.5.4. Convolutional Neural Network(CNN) :

Next comes the Convolutional Neural Network (CNN, or ConvNet) which is a class of deep neural networks which is most commonly applied to analyzing visual imagery.

Their other applications include video understanding, speech recognition and understanding natural language processing. Also, LSTM combined with Convolutional Neural Networks (CNNs) improved automatic image captioning like those are seen in Facebook. Thus you can see that RNN is more like helping us in data processing predicting our next step whereas CNN helps us in visuals analyzing. [8]

[8] https://monkeylearn.com/sentiment-analysis/

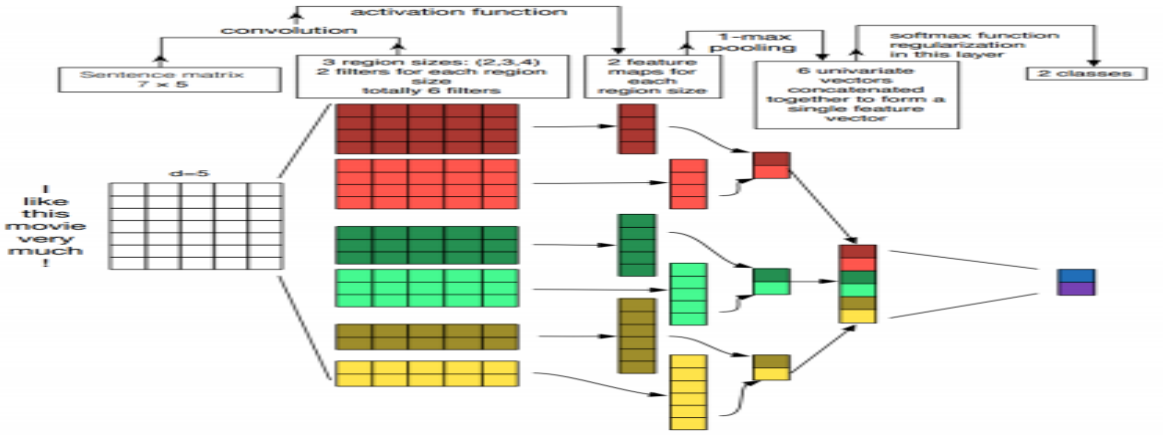


Figure.II.10. The CNN baseline architecture is as follows: "A CNN architecture for sentence categorization is depicted. We show three different filter region sizes: 2, 3, and 4, each with two filters. Filters execute convolutions on the sentence matrix and build (varying length) feature maps; 1-max pooling is applied to each map, which means that the greatest number from each feature map is saved. As a result, each of the six mappings generates a univariate feature vector, which is then concatenated to build a feature vector for the penultimate layer.The final softmax layer takes this feature vector as input and uses it to categorize the text; we're using binary classification here, so there are two potential output states." (Zhang, Y. and Wallace, B. C., 2015).

2.5.5. Gated Recurrent Units (GRUs)

GRU is capable of learning some dependencies which are introduced by Cho (Cho, et al.) . It is a special kind of RNN (Recurrent Neural Network). They enormously work on a large type of problems. They are one in every of the most fashionable, powerful and effective neural networks. This is expressly designed to avoid the long-term dependency drawback. GRU has fewer parameters than LSTM and so may train a bit quicker or needless iteration to generalize (Fischer, Thomas, and Christopher Krauss., 2018) .Rafal et al. showed that the GRU outperformed the LSTM on most tasks with the exception of language modeling . (Lecun, Yann, et al., 2015).

For every element of a sequence, GRU performs a similar task. That is the reason for which it is called recurrent. And the result being depended on the calculations of the previous. Another think about GRUs is that they have a unit called memory units. The calculated information is captured by it. In Figure. 11 a network, 'GRU' looks at xt input and outputs ht which is also the input of the next step. So a loop permits data to be allowed from one step to the next in the RNN.

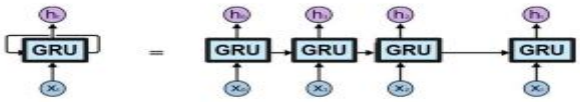


Figure.II.11. Unrolled GRUs

Because of the multiple copies of the same network, this neural network is called recurrent. Here each passing a message to the next network. Recurrent neural networks related to sequences and lists just like stock market data. And that chain-like nature is the reason behind it. To use for extracting the hidden pattern of the stock market, these networks have the natural architecture of the neural network

3. Transfer Learning (TL)

More recently, another paradigm shift in NLP towards using transfer learning approaches happened, as such have successfully been applied to many downstream tasks (Howard, J. and Ruder, S. , 2018) (Peters, 2018)A new neural network architecture called the transformer (Vaswani et al , 2017) has proven to be particularly useful. The most widely studied implementation of a transformer used for NLP transfer learning is a model called BERT (Bidirectional Encoder Representations from Transformers) that has been open-sourced by Google (Devlin et al., 2018)

3.1. Transformer

"Attention is All you Need" (Vaswani et al , 2017) It made numerous improvements to soft attention and allowed seq2seq modeling without the use of recurrent network units. Without using sequence-aligned recurrent architecture, the proposed "transformer" paradigm is entirely based on self-attention mechanisms.

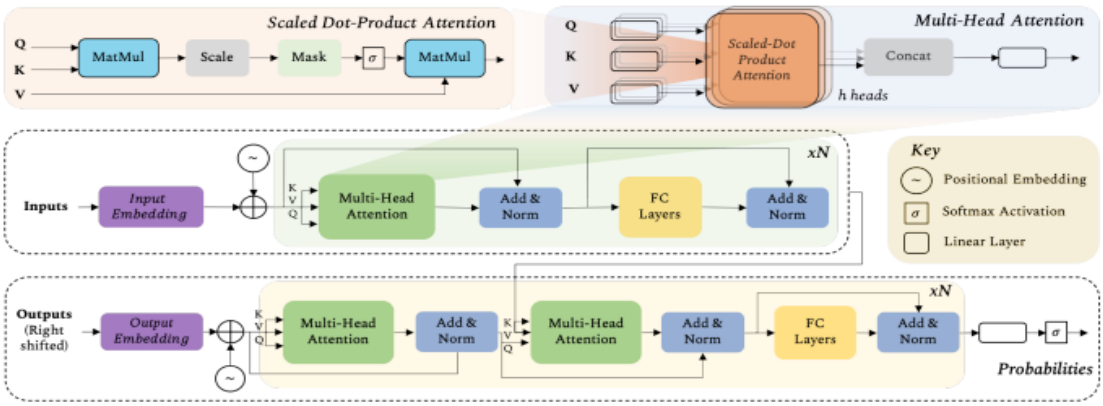


Figure.II.12. Architecture of the Transformer Model[9]

The model was first developed for the language translation task where an input sequence in one language is required to be converted to the output sequence in another language. The Transformer encoder (middle row) operates on the input language sequence and converts it to an embedding before passing it on to the encoder blocks. The Transformer decoder (bottom row) operates on

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in NeurIPS, 2017.

the previously generated outputs in the translated language and the encoded input sequence from the middle branch to output the next word in the output sequence. The sequence of previous outputs (used as input to the decoder) is obtained by shifting the output sentence to the right by one position and appending start-of-sentence token at the beginning. This shifting avoids the model to learn to simply copy the decoder input to the output. The ground-truth to train the model is simply the output language sequence (without any right shift) appended with an end-of-sentence token. The blocks consisting of multi-head attention (top row) and feed-forward layers are repeated N times in both the encoder and decoder.

3.1.1. Key, Value, and Query

The multi-head self-attention mechanism is the transformer's most important component. The encoded representation of the input is viewed by the transformer as a series of key-value pairs (K, V), each of dimension n (input sequence length); the keys and values are the encoder hidden states. In the decoder, the previous output is compressed into a query (Q of dimension m) and the next output is produced by mapping this query and the set of keys and values. (Vaswani et al , 2017)

The scaled dot-product attention is used by the transformer, the output is a weighted sum of the values, with the weight assigned to each value determined by the dot-product of the question with all keys:

$$\text{Attention}(Q, k, v)=\text{softmax}\left(\frac{Qk^T}{\sqrt{N}}\right)v$$

3.1.2. Self-attention

Self-Attention in Transformers Given a sequence of items, self-attention estimates the relevance of one item to other items (e.g., which words are likely to come together in a sentence). The self-attention mechanism is an integral component of Transformers, which explicitly models the interactions between all entities of a sequence for structured prediction tasks. Basically, a selfattention layer updates each component of a sequence by aggregating global information from the complete input sequence. Lets denote a sequence of n entities (x1, x2, · · · xn) by X ∈ R n×d , where d is the embedding dimension to represent each entity. The goal of self-attention is to capture the interaction amongst all n entities by encoding each entity in terms of the global contextual information. This is done by defining three learnable weight matrices to transform Queries (WQ ∈ R d×dq), Keys (WK ∈ R d×dk) and Values (WV ∈ R d×dv), where dq = dk. The input sequence X is first projected onto these weight matrices to get Q = XWQ, K = XWK and V = XWV . The output Z ∈ R n×dv of the self attention layer is,

$$Z = \text{softmax}\left(\frac{Qk^T}{\sqrt{N}}\right)v$$

For a given entity in the sequence, the self-attention basi-cally computes the dot-product of the query with all keys, which is then normalized using softmax operator to get the attention scores. Each entity then becomes the weighted sum of all entities in the sequence, where weights are given by the attention scores .

3.1.3. Multi-head Attention

Multi-Head Attention: In order to encapsulate multiple complex relationships amongst different elements in the sequence, the multi-head attention comprises multiple selfattention blocks (h = 8 in the original Transformer model [1]). Each block has its own set of learnable weight matrices {WQi,WKi,WVi }, where i = 0 · · ·(h−1). For an input X, the output of the h self-attention blocks in multihead attention is then concatenated into a single matrix [Z0, Z1, · · · Zh−1] ∈ R n×h·dv and projected onto a weight matrix W∈ R h·dv×d top row). (Vaswani et al , 2017) (Figure 26). The main difference of self-attention with convolution operation is that the filters are dynamically calculated in

stead of static filters (that stay the same for any input) as in the case of convolution. Further, self-attention is invariant to permutations and changes in the number of input points. As a result, it can easily operate on irregular inputs as opposed to standard convolution that requires grid structure. Furthermore, it has been shown in the literature how selfattention (with positional encodings) is theoretically a more flexible operation which can model the behaviour of convolutional models towards encoding local features[10]Cordonnier et al.[11]further studied the relationships between self-attention and convolution operations. Their empirical results confirm that multi-head self-attention (with sufficient parameters) is a more generic operation which can model the expressiveness of convolution as a special case. In fact, self-attention provides the capability to learn the global as well as local features, and provide expressivity to adaptively learn kernel weights as well as the receptive field (similar to deformable convolutions[12]).

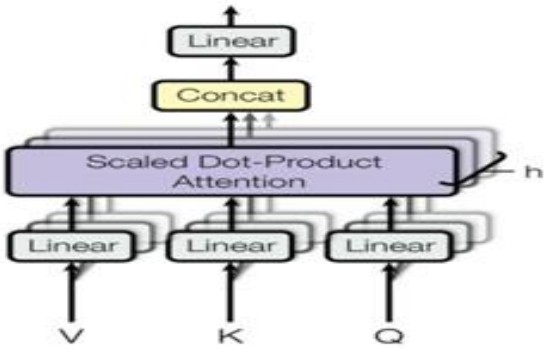


Figure.II.13. Multi-head scaled dot-product attention mechanism. (Vaswani et al , 2017)

Rather than only computing the attention once, the multi-head mechanism runs through the scaled dot-product attention multiple times in parallel. The independent attention outputs are simply concatenated and linearly transformed into the expected dimensions.

[10] J. Perez, J. Marinkovi ´ c, and P. Barcel ´ o, "On the turing complete- ´ ness of modern neural network architectures," in International Conference on Learning Representations, 2018.

[11] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," in International Conference on Learning Representations, 2019.

[12] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in Proceedings of the IEEE international conference on computer vision, pp. 764–773, 2017

3.1.4. Encoder

The encoder generates an attention-based representation capable of locating a specific piece of information from a potentially infinitely-large context.

The encoder is made up of N = 6 identical layers stacked on top of each other. Each layer is divided into two sub-layers. The first is a multi-head self-attention system, while the second is a basic, fully linked feed-forward network based on location. After layer normalization, they use a residual connection around each of the two sub-layers. LayerNorm(x + Sublayer(x)) is the output of each sub-layer, where Sublayer(x) is the function implemented by the sub-layer itself. All sub-layers in the model, as well as the embedding layers, provide outputs of size dmodel = 512 to support these residual connections. (Vaswani et al , 2017)

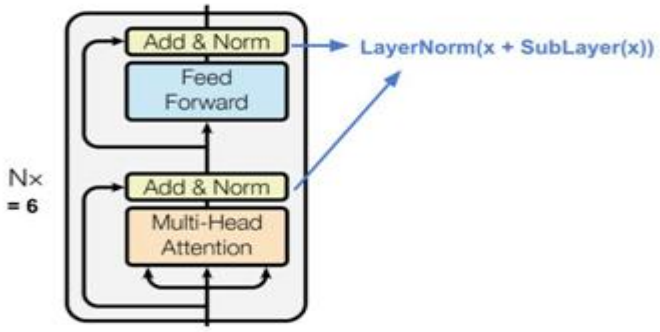


Figure.II.14. The transformer's encoder. (Vaswani et al , 2017)

3.1.5. Decoder

A stack of N = 6 identical layers also makes up the decoder. The decoder adds a third sub-layer to each encoder layer, which conducts multi-head attention over the encoder stack's output, in addition to the two sub-layers in each encoder layer. We use residual connections surrounding each of the sub-layers, followed by layer normalization, much like the encoder. They also change the decoder stack's self-attention sub-layer to prevent positions from attending to following positions. Because of the masking, as well as the fact that the output embeddings are offset by one position, the predictions for position I can only be based on the known outputs at positions less than i (Vaswani et al , 2017).

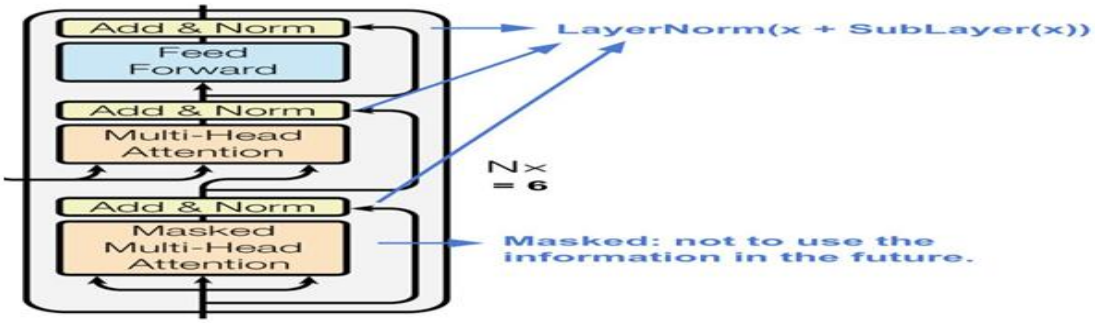


Figure.II.15. The transformer's decoder (Vaswani et al , 2017).

3.2. Recent Language Understanding Models Based on Transfer Learning

The growing trend of employing Transfer Learning to pre-train deep learning architectures allows for the creation of a wide range of language understanding models. For various NLP tasks, these models have shown a significant boost in performance. Some of these models are mentioned below:

Today, transfer learning is at the heart of language models like Embeddings from Language Models (ELMo) and Bidirectional Encoder Representations from Transformers (BERT) — which can be used for any downstream task.

3.2.1. Embeddings from Language Models (ELMo)

ELMo ("Embeddings from Language Model") is a word embedding method for representing a sequence of words as a corresponding sequence of vectors. Character-level tokens are taken as the inputs to a bi-directional LSTM which produces word-level embeddings.

Embeddings from Language Models, or **ELMo**, is a type of deep contextualized word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large text corpus.

A biLM combines both a forward and backward LM. ELMo jointly maximizes the log likelihood of the forward and backward directions. To add ELMo to a supervised model, we freeze the weights of the biLM and then concatenate the ELMo vector $ELMO_k^{task}$ with xk and pass the ELMO enhanced representation $[x_k; ELMO_k^{task}]$ into the task RNN. Here x_k is a context-independent token representation for each token position. (Peters et al., 2018)

3.2.2. Bidirectional Encoder Representation from Transformers (BERT)

BERT (Bidirectional Encoder Representations from Transformers) is a bidirectional Transformer architecture that builds upon previous work in pretraining contextual language representations - including Semi-supervised Sequence Learning (Dai and Le, 2015), GPT (Generative Pretraining) (Radford et al., 2016), ELMo (Peters et al., 2018), and ULMFiT (Howard and Ruder, 2018). As indicated by its name, BERT uses only the encoder part of the Transformer. When published in October 2018 by a research team from Google AI Language, BERT introduced two novel tasks for self-supervised pretraining: Masked Language Modeling (Masked LM) and Next Sentence Prediction; as well as representations that are jointly conditioned on both left and right context in all layers (Devlin et al., 2018). Subsequently, the model can be fine-tuned for a specific task

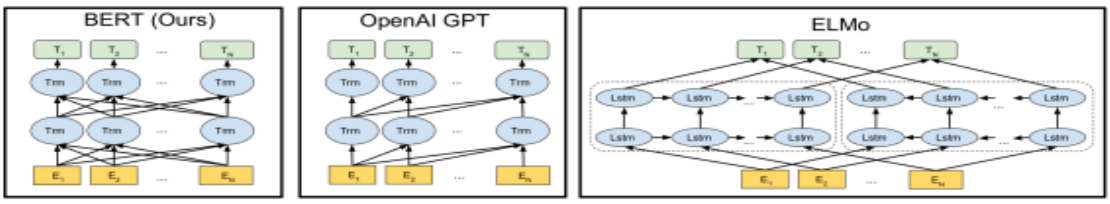


Figure.II.16. Differences in pretraining model architectures. (Devlin et al., 2018)

A bidirectional transformer is used by BERT. A left-to-right transformer is used by OpenAI GPT. ELMo generates features for downstream tasks by concatenating left-to-right and right-to-left LSTM that have been trained individually. Only BERT representations in all levels are jointly conditioned on both left and right context among the three.(Devlin et al., 2018)
as classifying sentences, sentence pairs, or tokens. In contrast to classical word embedding models like GloVe or Word2Vec, BERT uses a limited vocabulary (around 30,000 words, compared to 400,000 words used by GloVe). The reason for this is that BERT uses WordPiece tokenization (Schuster, M. and Nakajima, K., 2012)which splits the words into subword units. When pretraining the BERT language model, the Masked LM task and the Next Sentence Prediction task are jointly trained, with the objective of minimizing the combined loss function of both.

A. Masked LM (MLM)

Unidirectional models are effectively trained by predicting each word in the sentence based on the preceding words. However, training bidirectional models by merely conditioning each word on its preceding and subsequent words is not practicable since the predicted word would indirectly "see itself" in a multi-layer model. The researchers devised a simple solution for this: random masking of some of the words in the input and bidirectional conditioning of each word to predict the masked words.
Despite the fact that this concept has been known for a long time (Taylor, 1953), BERT was the first to effectively apply it to pretrain a deep neural network. So, based on the context supplied by the other, non-masked, words in the sequence, the challenge is to guess the original value of the masked words; an example is illustrated in Figure 33. The [MASK] token is used to replace 15% of the words in each sequence.
The "masked" word's true value is picked at random from a data generator: 80% of the time, replace the word with "[MASK]", 10% of the time, replace the word with a random word, and 10% of the time, leave the word alone. This causes the Transformer encoder to keep track of every input token's distributional context, and BERT learns to predict missing words wherever in a sentence. To do so, it must first comprehend many of the key word connections. This information can then be used to other tasks (transferred).

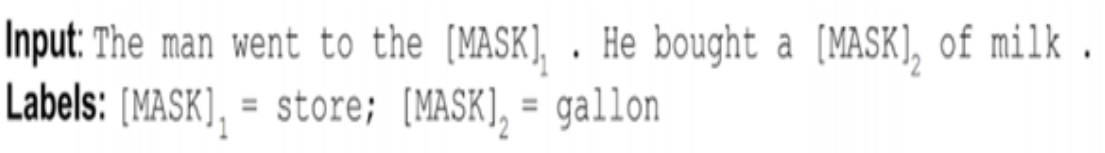


Figure.II.17. Masked LM example (Devlin and Chang, 2018)

The architecture for Masked LM is depicted in Figure 17. Technically speaking, the prediction of the output words involves: (1.) An additional classification layer on top of the encoder output. (2.) Multiplication of the output vectors by the embedding matrix, thereby transforming them into

the vocabulary dimension. (3.) A calculation of the probability for each word in the vocabulary with softmax. The BERT loss function considers only the prediction of the masked words and ignores the prediction of the non-masked words. As a result, the model converges more slowly than directional models, a characteristic that is compensated by the increased context awareness (Horev, 2019)

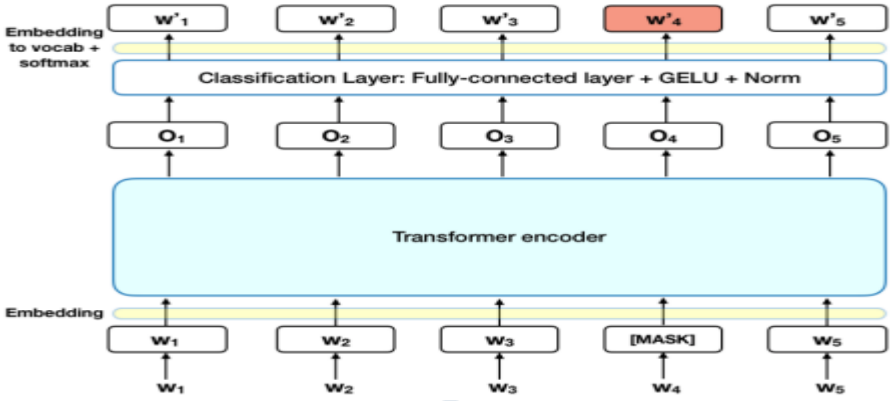


Figure.II.18. Masked LM architecture of BERT (Horev, 2019)

B. Next Sentence Prediction (NSP)

BERT also learn relationships between multiple sentences by pretraining on an additional task: Given two sentences A and B, is B the actual next sentence that comes after A in the corpus or is it just a random sentence? See Figure 19 for an example. During training 50% true pairs and 50% false pairs are selected at random.



Figure.II.19. Next sentence prediction example (Devlin and Chang, 2018) Given two sentences (A and B), the task is to learn how likely it is that B follows A. This is included in the pretraining process, to improve BERT's capabilities of handling relationships between multiple sentences.

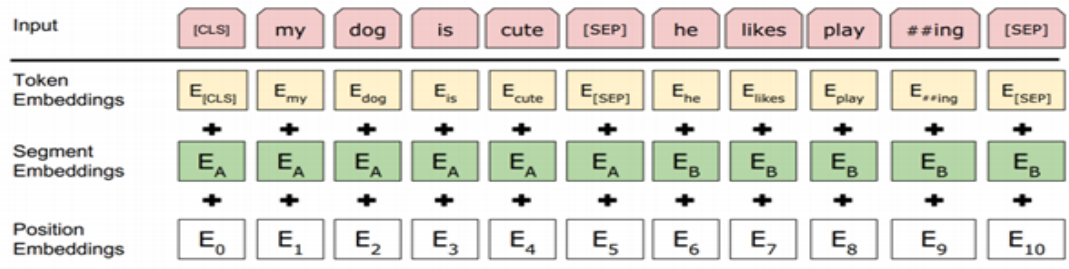


Figure.II.20 . BERT input representation: the input embedding is the sum of the token embeddings, the segmentation embeddings and the position embeddings(Devlin et al., 2018)

Figure 20. provides insight into how BERT's Next Sentence Prediction works. To enable the model to differentiate between the two sentences during training, the inputs are
processed as follows before entering the model: (1.) A [CLS] token is added at the beginning of the first sentence and an [SEP] token is added at the end of each sentence. (2.) For each token a sentence embedding is added, that indicates whether the token belongs to Sentence A or Sentence B. Sentence embeddings are conceptually similar to token embeddings with a size two vocabulary. (3.) Furthermore, for each token a positional embedding is added, that indicates its position in the sequence. The concept and the implementation of the positional embedding stem from the Transformer paper.
Masked LM and Next Sentence Prediction are coupled for training the BERT model, to minimize the combined loss function of the two techniques.

4. Ensemble Learning

Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem.
It is a powerful method to improve the performance of the model. It combines the predictions from multiple models. The results are improved by using this ML algorithm.
Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one. Other applications of ensemble learning include assigning a confidence to the decision made by the model, selecting optimal (or near optimal) features, data fusion, incremental learning, nonstationary learning and error-correcting. This article focuses on classification related applications of ensemble learning, however, all principle ideas described below can be easily generalized to function approximation or prediction type problems as well.

5. Attention Layer

In neural networks, attention is a technique that mimics cognitive attention. The effect enhances some parts of the input data while diminishing other parts — the thought being that the network should devote more focus to that small but important part of the data. Learning which part of the data is more important than others depends on the context and is trained by gradient descent.
A neural network is considered to be an effort to mimic human brain actions in a simplified manner. Attention Mechanism is also an attempt to implement the same action of selectively concentrating on a few relevant things while ignoring others in deep neural networks.

5.1 Attention Mechanism

A mechanism that can help a neural network to memorize long sequences of the information or data can be considered as the attention mechanism and broadly it is used in the case of Neural machine translation(NMT). As we have discussed in the above section, the encoder compresses

the sequential input and processes the input in the form of a context vector. We can introduce an attention mechanism to create a shortcut between the entire input and the context vector where the weights of the shortcut connection can be changeable for every output.

Because of the connection between input and context vector, the context vector can have access to the entire input, and the problem of forgetting long sequences can be resolved to an extent. Using the attention mechanism in a network, a context vector can have the following information:

- Encoder hidden states.
- Decoder hidden states.
- Alignment between source and target.

Using the above-given information, the context vector will be more responsible for performing more accurately by reducing the bugs on the transformed data.

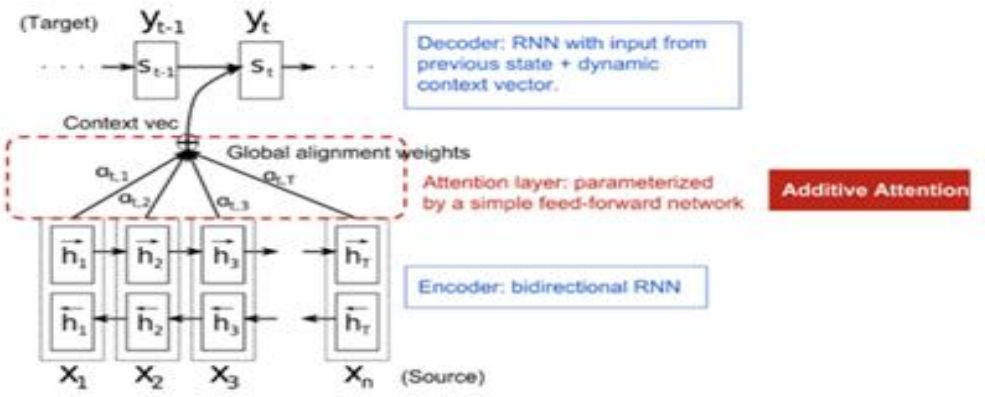


Figure.II.21. The encoder-decoder model with additive attention mechanism in (Bahdanau, Cho, & Bengio).

The above given image is a representation of the seq2seq model with an additive attention mechanism integrated into it. Let's introduce the attention mechanism mathematically so that it will have a clearer view in front of us. Let's say that we have an input with n sequences and output y with m sequence in a network. $x = [x_1, x_2, \ldots, x_n]$

$$y = [y_1, y_2, \ldots, y_m]$$

Now the encoder which we are using in the network is a bidirectional LSTM network where it has a forward hidden state and a backward hidden state. Representation of the encoder state can be done by concatenation of these forward and backward states.

$$h_i = [\overrightarrow{h}_i^{\top}; \overleftarrow{h}_i^{\top}]^{\top}, i = 1, \ldots, n$$

Where in the decoder network, the hidden state is

$$s_t = f(s_{t-1}, y_{t-1}, \mathbf{c}_t)$$

For the output word at position *t*, the context vector C_t can be the sum of the hidden states of the input sequence.

$$\begin{aligned}
\mathbf{c}_t &= \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i && \text{; Context vector for output } y_t \\
\alpha_{t,i} &= \text{align}(y_t, x_i) && \text{; How well two words } y_t \text{ and } x_i \text{ are aligned.} \\
&= \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))} && \text{; Softmax of some predefined alignment score..}
\end{aligned}$$

Here we can see that the sum of the hidden state is weighted by the alignment scores. We can say that $\{\alpha_{t,i}\}$ are the weights that are responsible for defining how much of each source's hidden state should be taken into consideration for each output.

5.1.1 A Family of Attention Mechanisms

With the help of the attention, the dependencies between source and target sequences are not restricted by the in-between distance anymore, Given the big improvement by attention in machine translation, it soon got extended into the computer vision field (Xu, et al., 2015) and people started exploring various other forms of attention mechanisms (Luong, Pham, & Manning, 2015) (Britz, Goldie, Luong, & Le, 2015); (Vaswani et al , 2017)

5.1.2. Summary

Table.II.1.A summary table of several common attention mechanisms and their associated alignment score functions.

There can be various types of alignment scores according to their geometry. It can be either linear or in the curve geometry. Below are some of the popular attention mechanisms:

Name	Alignment score function
Content-base attention (Graves & Wayne, Neural Turing Machines, 2014)	$score(s_t, h_i) = cosine[s_t, h_i]$
Additive(*) (Bahdanau, Cho, & Bengio, 2015)	$score(s_t, h_i) = v_a^t \tanh(w_a[s_t, h_i])$
Location-base (Luong, Pham, & Manning, 2015)	$\alpha_{t,i} = softmax(w_a s_t)$ This simplifies the softmax alignment to only depend on the target position .

General (Luong, Pham, & Manning, 2015)	$score(s_t, h_i) = s_t^T W_a h_i$ Where W_a is a trainable weight matrix in the attention layer .
Dot-Product (Luong, Pham, & Manning, 2015)	$score(s_t, h_i) = s_t^T h_i$
Scaled Dot-Product(^) (Vaswani et al , 2017)	$score(s_t, h_i) = \frac{s_t^T h_i}{\sqrt{n}}$ Note : very similar to the dot-product attention except for a scaling factor ;where n is the dimension of the source hidden state .

5.1.3. Self-Attention

When an attention mechanism is applied to the network so that it can relate to different positions of a single sequence and can compute the representation of the same sequence, it can be considered as self-attention and it can also be known as intra-attention. In the paper about. Long Short-Term Memory-Networks for Machine Reading by Jianpeng Cheng, Li Dong, and Mirella Lapata, we can see the uses of self-attention mechanisms in an LSTM network. The below image is a representation of the model result where the machine is reading the sentences.

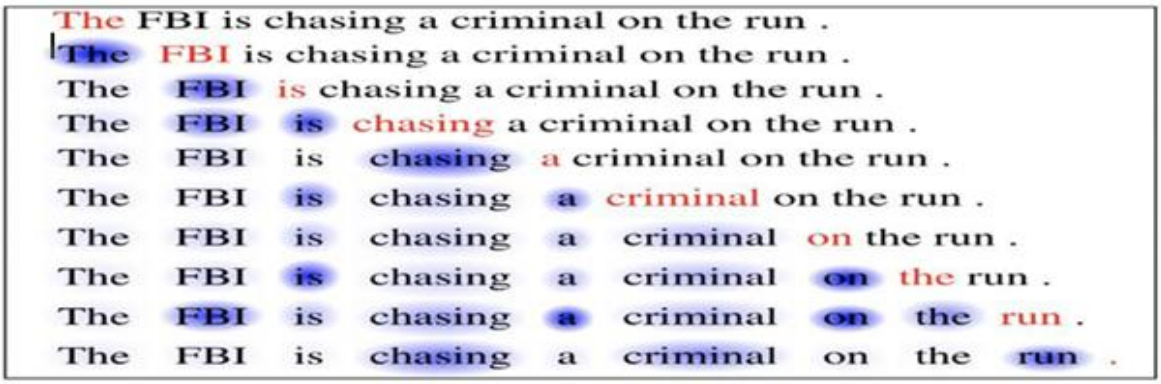


Figure.II.22. The current word is in red and the size of the blue shade indicates the activation level. (cheng et al , 2016)

Here in the image, the red color represents the word which is currently learning and the blue color is of the memory, and the intensity of the color represents the degree of memory activation.
As of now, we have seen the attention mechanism, and when talking about the degree of the attention is applied to the data, the soft and hard attention mechanism comes into the picture, which can be defined as the following.

5.1.4. Soft vs Hard Attention

The attention mechanism is applied to images to generate captions. To extract features, the image is first encoded by a CNN. The convolution features are then consumed by an LSTM decoder, which produces descriptive words one by one, with the weights acquired through attention (Luong, Pham, & Manning, 2015); (Bahdanau, Cho, & Bengio, 2015).

5.1.5. Global vs Local Attention

(Luong, Pham, & Manning, 2015) proposed the "global" and "local" attention The global attention is close to soft attention, while the local attention is an unusual combination of hard and soft, an improvement over hard attention to make it distinguishable: the model predicts a single aligned position for the current target word first and then computes a context vector using a window-based around the source position.

6. Embedded Vectors

Most machine learning methods that use approximation to translate input to output require numerical representation of input data. Embedding techniques (also known as vectorizing or encoding) turn input data (words, phrases, paragraphs, documents, dates, emoji, graphs, and so on) into real numbers, capturing the underlying semantic relationship between the input data. Embedding models are a prominent deep learning-based NLP job and are one
of the effective uses of unsupervised learning. Word embeddings were first introduced by (Chorowski et al ., 2015). There have been some notable models that can be utilized to represent the input text mentioned. (Dessí et al ., 2020).

6.1. Collobert and Weston (C&W) model

C&W model proposed in (Collobert et al., 2011) has been designed using multi-layered neural network architecture, trained on large dataset and carries syntactic and semantic meaning. This model is designed agnostic to any task-specific feature engineering and therefore serves as useful word representation model for wide variety of NLP tasks.

6.2. Word2Vec

Word2vec is a technique for natural language processing published in 2013. The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. As the name implies, word2vec represents each distinct word with a particular list of numbers called a vector. The vectors are chosen carefully such that a simple mathematical function (the cosine similarity between the vectors) indicates the level of semantic similarity between the words represented by those vectors. (Mikolov et al , 2013)

Word2vec is a popular technique for modelling word similarity by creating word vectors. It's a method that uses neural networks to model word-to-word relationships. Basically, the algorithm takes a large corpus of text as input and produces a vector, known as a context vector, as output.

Word2vec starts with a simple observation: words which occur close together in a document are likely to share semantic similarities. For example, “king” and “queen” are likely to have similar meanings, be near each other in the document, and have related words such as “man” or “woman.” Word2vec takes this observation and applies it to a machine learning algorithm:

Skip-gram: works well with a small amount of the training data, represents well even rare words or phrases .
CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words.

6.3. FastText

In 2016, artificial neural nets had gained quite some traction, and Word2Vec has proven its usefulness in many areas of NLP. However, there was one unsolved problem: generalization to unknown words. **FastText** — a development by Facebook released in 2016 — promised to overcome this obstacle. FastText, by Facebook Research, is a library for efficient learning of word representations and text classification. FastText supports supervised (classifications) and unsupervised (embedding) representations of words and sentences.

6.4. Global Vectors for Word Representation (GloVe)

GloVe, coined from Global Vectors, is a model for distributed word representation. The model is an unsupervised learning algorithm for obtaining vector representations for words. This is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity. (Abad et all , 2016)Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. It is developed as an open-source project at Stanford and was launched in 2014. As log-bilinear regression model for unsupervised learning of word representations, it combines the features of two model families, namely the global matrix factorization and local context window methods. (Kalajdziski, 2018)

6.5. Graphs from LOw-level unit Modeling (GLoMo)

Unsupervisedlatent graph learning is used to create graphs from the low-level unit modeling (GLoMo) framework (Yang et al, 2018)It's also a transfer learning system designed to enhance sentiment analysis, natural language inference, question answering, and image classification efficiency in NLP tasks.

6.6. Universal Language Model Fine-tuning (ULMFiT)

ULMFiT is a transfer learning model that can be applied to any task involving natural language processing. For sentiment analysis, ULMFiT's pre-trained models can be used. A language model is pre-trained on a general domain before being fine-tuned on the target domain
In this process. It claims to be universal since its operation is independent of document size, number, and mark. It uses the same architecture and training for all activities and does not include domain-specific documentation and labels.

Part2:

1. Related work:

This section presents recently proposed methods for detecting offensive speech and hate from user-generated content on social media.

1.1. Current state in Hate Speech Detection and Related Concepts:

There are some researches that have discussed different related terminologies which serves similar related concept to the phenomena of hate speech (e.g. cyber-bullying, abusive language, radicalization detection). The analysis of these different terminologies will definitely help to reach insights from different perspectives in current situation and will also contribute in spotting and recognizing the interrelationship among these terminologies.

1.1.2. Abusive language detection

It is a broad concept that encompasses all abusive language. Hate speech falls under the category of abusive words. Profanity is also included in this vocabulary (the use of inappropriate words). Many studies, however, refer to abusive language as objectionable language. To detect offensive language, (Y. Chen et al, 2012)used YouTube comments as a dataset. They employed a combination of lexical and grammatical data, as well as the user's writing style, to forecast future behavior. Furthermore, Wiegand et al. expected that they would be able to distinguish abusive phrases from negative polar expressions.. They used a base lexicon to select a limited subset of negative polar expressions, and then used crowd sourcing to classify the offensive phrases. . (G. Xiang et al, 2012) proposed a similar approach to detect offensive content in Twitter.. Their characteristics were based mostly on the linguistic regularities of profane phrases, as well as statistical topic modeling on a large dataset. Park and Fung used the dataset provided by to compare the performance of one-step and two-step classifiers in a deep learning scenario (Z. Waseem and D. Hovy, 2016). They believe that combining two classifiers (for example, CNN and logistic regression) can improve performance. Furthermore, (Y. Chen et al, 2012)employed FastText as a neural network classifier to detect abusive text across a variety of social media platforms. They discovered that FastText performs worse than SVM as a classifier.

1.1.3. Cyber bullying detection

The electronic form of traditional bullying is called cyberbullying. which is the aggression and harassment that is targeted to an individual who is unable to defend himself Bullying is known with its repetitive act to the same individual, unlike hate speech which is more general and not necessarily intended to hurt a specific individual. (Dinakar et al, 2011)research is one of the pioneers and most cited researches for the textual cyberbullying detection. Their experiment was based on a corpus of 4500 YouTube comments. Their result showed that showing the polarities

of the dataset outperformed categorizing the dataset into a multiclass. Both of (Nahar et al, 2014)and (M. Di Capua et al, 2016)presented unsupervised approach for cyberbullying detection. (Özel et al, 2017) work is unique to this area because they have investigated Turkish language in order to detect cyberbullying from twitter and Instagram text. Their results showed that Naïve Bayes Multinational showed the best results in both accuracy and total training and testing time. Finally, (Pawar et al, 2018)utilized distributed computing for cyberbullying detection. Their work focuses mostly on the robust performance rather than the accuracy alone.

1.1.4 . Radicalization detection

This concept is usually referred to as a motive towards violent extremism. Usually radical groups have an ideology that considers violence as a legitimate action when it serves to address their concerns. Radicalization and hate speech are closely related and usually mentioned as if they have the same meaning but actually radicalization comes under hate speech as it has specific tendencies towards religious believes. (P. Wadhwa and M. P. S. Bhatia, 2013)referred to radical groups as "cyber-extremists". They investigated the possibility of the detection of such act in Twitter using unsupervised approach. They came with the conclusion that fully unsupervised approach will not be able to detect the right topics for this issue, manual intervention is necessary to reach better results because tweets have a dynamic nature. (S. Agarwal and A. Sureka, 2015)introduced a semi supervised approach to detect radicalization in twitter. They had a mixture of labeled and unlabeled data.

1.1.5. Hate speech detection

A hate speech detection approach established a lexical baseline for discriminating between hate speech and profanity on a standard dataset (Zampieri et al., 2019). In (Badjatiya et al., 2017) a Twitter corpus of 16K texts for hate speech identification was manually annotated, and for which various DL approaches have been evaluated. Another research using DL approaches has been proposed for hate speech detection in Indonesian Language (Sutejo and Lestari, 2018). The authors evaluated different feature models, where textual features produced promising results (87.98% of F1- score). Some studies have been carried out in Arabic language. For instance, an approach for detecting cyberbullying in Arabic texts has been proposed by (Haidar et al.2017), where the approach focused on preventing cyberbullying attacks. In particular, it uses NLP techniques to identify and process Arabic words, and ML classifiers to detect the bullying content.

1.2. Arabic Hate Speech Detection

A limited number of Arabic researches have contributed to that particular area. In the other hand, many Arabic researches were investigated in similar areas which we can call "Anti-social behaviors" such as, Abusive or offensive language and cyberbullying.

1.2.1. Arabic anti-social behaviour detection

Starting with Abusive language detection. Abozinadah paved the way in this area and contributed in three researches tailored for this area. First (Abozinadah et al, 2015)proposed a model in response to Arab governments needs of blocking such abusive contents. They created their own test set and made it publicly available. Then in (E. A. Abozinadah and J. H. Jones, 2016) enhanced the previous work by proposing a lexicon that is fed by an Arabic word correction method to enhance the detection of such abusive words. A third work by Abozinadah is which used statistical learning approach for the detection process to overcome the limitation in the BOW approach presented in other previous works. . (Mubarak et al, 2017) mubarak al work aimed to build a a large scale corpus of Arabic tweets that are classified to (Obscene, offensive and clean) and made it available for next researchers. Another two contributions by (A. Alakrot, 2018)((A. Alakrot, L. Murray, and N. S. Nikolov,, 2018). In the first work, they have constructed a corpus of Arabic comments from YouTube and made it publicly available for abusive detection purposes. In their second work, an empirical examination of the dataset has been performed. They concluded that a combining N-gram and stemming may results in lower performance. Alshehri et al. followed the same path of Abozinadah and Azalden but the concept is slightly different. They created a large scale of adult content in Arabic Twitter. Consequently, a large lexicon was built based on that corpus. In addition to the previous behaviors, cyberbullying is another serious issue that has been addressed by many researchers in English language. For the Arabic language, (Haidar et al, 2017)made the first attempt to detect cyberbullying in Arabic language. Their work was the first step into this area, since it needs a lot of enhancements such as considering more features related to cyberbullying and choosing better feature representation. (A. H. Alduailej and M. B. Khan, 2017)discussed the main challenges of detecting cyberbullying in Arabic language. The fundamental challenge was that we need to discover the context before deciding whether it is considered cyberbullying or not.

1.2.2. Arabic hate speech detection

Hate speech detection in English has been extensively researched by a lot of authors who explored all forms of hate speech (racial, sexism, religious and general hate). In contrast, the Arabic language contains little resources for recognizing various types of hate speech. (Albadi et al, 2018)were the first to address the issue of religious hatred on Arabic Twitter, but they were not confronted with the other categorizations of hate speech. They created and graded a vocabulary of the most often used religious phrases. For this challenge, they tried a variety of classifiers, including GRU RNN, which outperformed the other classifiers. They explained why they chose GRU over LSTM, claiming that GRU performs better with smaller datasets and is quicker in terms of training time, as well as having a reduced likelihood of overfitting small datasets.

Algerian dialect hate speech detection:

It is noted that a few studies have been conducted with Arabic language. However, only a limited number of researches have focused on hate speech detection in Algerian dialect. (Guellil et al, 2021)Developed the first corpus in Algerian for hate speech detection against women in Arabic community on social media, specifically YouTube. This corpus contains 373984 comments, then they created two datasets. The first is imbalanced, but the second is perfectly balanced. They used two different algorithms for features extraction which are, Word2vec with classic methods and FasText with Deep learning methods. for the classic classification methods, they used: GaussianNB (GNB), LogisticRegression (LR), RandomForset (RF), SGDClassifier (SGD, with loss='log' and penalty='l1') and LinearSVC (LSVC with C='1e1'). for the deep learning classification, they used Convolutional Neural Network (CNN), long short-term memory (LSTM) network, and Bi-directional LSTM(Bi-LSTM) network, for each model they used six layers, the first layer is a randomly-initialised word embedding layer where the weights of the embedding_matrix are calculated using fasText (with both SG and CBOW implementation). The next layer is the CNN/ LSTM/BiLSTM layer that scans the feature map (depending on the model). Then the global max pooling layer is applied to reduce the dimensionality of the CNN/LSTM/BiLSTM representations. a Dropout layer with a probability equal to 0.5 is added. After that, the results are fed into a single feed-forward (fully-connected) layer with Relu activation. Finally, the output of that layer goes through a sigmoid layer that predicts the output classes. As a result, for the unbalanced corpus, the F1-score obtained with SG is up to 0.86 for CNN, 0.85 for LSTM and 0.83 for Bi-LSTM. whereas with CBOW 0.83 for CNN, 0.84 for LSTM, 0.82 for Bi-LSTM. For the balanced corpus, the F1-score obtained with SG is 0.82 for CNN, 0.80 for LSTM, 0.85 for Bi-LSTM. whereas with CBOW 0.77 for CNN, 0.79 for LSTM, 0.79 for Bi-LSTM.

Summary and Analysis

The following tables summarize all of the articles that have been discussed and are organized by time series. These tables cover the following topics in order: Antisocial conduct in English, hate speech in English, and antisocial behavior in Arabic are all examples of antisocial behavior. These tables can be used as a quick reference for all of the key work in social media automated detection that has been done. All of the approaches are simply presented, as are the results of their numerous investigations. All of the hate speech phrases and their contributions are included in Tables 3,4,5. All multilingual contributions and publications dealing with hate speech are included in Table 6.

Finally, table 7 highlights the Arabic language by compiling all articles on the detection of anti-social conduct on social media platforms. The Algerian dialect contributions are shown in Table 7. In the results column, the best findings from each study are highlighted.

Table.II.2. Summary of the current state of anti-social behaviour detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F).

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
Abusive Language (English)								
(Y. Chen et al., 2012)	2012	Youtube	Un-Supervised	Lexical and syntactic	match rules	0.98	0.94	-
(Xiang et al., 2012)	2012	Twitter	Semi-Supervised	Topic modelling	Logistic regression	-	-	0.84
(Park & Fung, 2017)	2017	Twitter	Supervised	Character and Wod2vec	Hybrid CNN	0.71	0.75	0.73
(H. Chen et al., 2017)	2017	Youtube, Myspace, SlashDot	supervised	word embeddings	FastText	-	0.76	-
(Wiegand et al., 2018)	2018	Twitter, Wikipedia, UseNet	Supervised	Lexical, linguistics and word embedding	SVM	0.82	0.80	0.81

Table.II.3. Summary of the current state of cyberbullying detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F).

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
Cyberbullying (English)								
(Dinakar et al., 2011)	2011	YouTube	Supervised	tf-df, lexicon,	PoS tag, bigram	SVM	0.66	-
(Nahar et al., 2014)	2014	Myspace, Slashdot	Semi-Supervised	Linguistic features	Fuzzy SVM	0.69	0.82	0.44
(Capua et al., 2016)	2016	YouTube, Form-Spring, Twitter	Un-Supervised	Semantic and syntactic features	GHSOM network and K-mean	0.60	0.94	0.74

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
(Pawar et al., 2018)	2018	Form-spring	Supervised	Bag of words	M-NB and Stochastic Gradient Descent	-	-	0.90
Cyberbullying (Turkish)								
(Özel et al., 2017)	2017	Twitter, Instagram	Supervised	Bag of words	M-Naïve Bayes	-	-	0.79
Radicalization (English)								
(Wadhwa & Bhatia, 2013)	2013	Twitter	Un-Supervised	Topic identification, N-grams	Topic-entity mapping	-	-	-
(S. Agarwal & Sureka, 2015)	2015	Twitter	Semi-Supervised	Linguistic, Term Frequency	LibSVM	-	-	0.83
(Fernandez & Alani, 2018)	2018	Twitter	Supervised	Semantic Context	SVM	0.85	0.84	0.85

Table.II.4. Summary of the current state of religion hate speech detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F).

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
Religious hate speech (English)								
(Warner & Hirschberg, 2012)	2013-Yahoo newsgroup	not antiSemitic, not antiitic Anti-Semitic, not anti-Semitic	supervised	Template-based, PoS tagging	SVM	0.59	0.68	0.63
Racial hate speech (English)								
(Huang et al., 2018)	2013-Twitter	Racist, Non-racist	Supervised	Unigram	Naïve Bayes	-	-	-
General hate speech (English)								
(Burnap & Williams, 2014)	2014-twitter	Yes,no	supervised	BOW, Dependencies, Hateful Terms	Bayesian Logistic Regression	0.89	0.69	0.77
(Gitari et al., 2015)	2015- Blog	No hate,Weakly hate,Strongly hate	SemiSupervised	Lexicon, Semantic, theme-based features	Rule based	0.73	0.68	0.70
(Le & Mikolov, 2014)	2015-Yahoo Finance	Hateful, Clean	Supervised	Paragraph2vec, CBOW	Logistic regression	-	-	-
(Waseem & Hovy, 2016)	2016-Twitter	Hate, not hate	Supervised	Character n-grams	Logistic regression	0.72	0.77	0.73
(Watanabe et al., 2018)	2018-twitter	Hateful, Offensive, Clean	Supervised	Sentiment-Based semantic, unigram	J48graft	0.79	0.78	0.78

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
(Malmasi & Zampieri, 2018)	2018-Twitter	Hate, Offensive, ok	Supervised	N-grams, Skip-grams, Hierarchical Word clusters	RBF kernel SVM	0.78	0.80	0.79
(Gambäck & Sikdar, 2017)	2017-twitter	Non-hate, racism, sexism, both	Supervised	Character N-grams, word2vec	CNN	0.85	0.72	0.78
(Joulin et al., 2017)	2017-twitter	Sexist, racist, neither sexist nor racist	Supervised	Random Embedding	LSTM and GBDT	0.93	0.93	0.93
(Pitsilis et al., 2018)	2018-twitter	Neutral, racism or sexism	Supervised	Word-based frequency vectorization	RNN and LSTM	0.90	0.87	0.88
(Zhang et al., 2018)	2018-twitter	Racism, sexism, both, non-hate	Supervised	Word embedding	CNN+GRU	-	-	0.94

Table.II.5. Summary of the Italian,Indonesia and German contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F).

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
General hate speech (Italian)								
(Del Vigna et al., 2017)	2017-Face- book	Hate, Not hate	Supervised	Morpho-syntactical, sentiment polarity, word embedding lexicons.	SVM RNN and LSTM	0.75 0.70	0.68 0.75	0.71 0.72
General hate speech (Indonesian)								
(Alfina et al., 2017)	2017-Twitter	Hate speech /Non hate	Supervised	BOW and n-gram	RandomForest decision tree	-	-	0.93

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
General hate speech (German)								
(Jaki & De Smedt, 2019)	2018-twitter	Muslim, Terrorist, Islamo-Fascistoid	Un- Supervised	Skip grams and Character trigrams	K-means, single-layer averaged Perceptron	0.84	0.83	0.84

Table.II.6. Summary of the Arabic contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F).

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
Abusive language (Arabic)								
(Abozinadah et al., 2015)	2015-Twitter	Abuser, Normal	Supervised	Profile and tweet-based features, bag of words, N-gram, TF-IDF	Naïve Bayes	0.85	0.85	0.85
(A. Abozinadah & H. Jones, Jr, 2016)	2016-Twitter	Abusive, Legitimate Accounts	Un-Supervised	Lexicon, bag of words (BOW), N-gram	SVM	0.96	0.96	0.96
(Abozinadah & Jones, 2017)	2017-Twitter	Non-Abusive, Abusive	Supervised	PageRank (PR) algorithm, Semantic Orientation (SO) algorithm, statistical measures.	SVM	0.96	0.96	0.96
(Mubarak et al., 2017)	2017-Twitter, Arabic News Site	Obscene, Offensive and Clean	Un- supervised	unigram and bigram, Log Odds Ratio (LOR), Seed Words lists	None. Just performed extrinsic evaluation	0.98	0.45	0.60

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
(Alakrot et al., 2018b, 2018a)	2018- YouTube	Offensive, In- Offensive	Supervised	N-gram	SVM	0.88	0.80	0.82
Violent content (Arabic)								
(E Abdelfatah et al., 2017)	2017- Twitter	Violent, Non violent	Un- supervised	Sparse Gaussian process latent variable model, morphological features, Vector Space Model	K-means clustering	0.56	0.60	0.58
Adult content (Arabic)								
(Lefever et al., 2018)	2018- Twitter	Adult, Regularuser	Supervised	Lexicon, N- grams, bag-of means(BOM)	SVM	0.70	0.93	0.78
Cyberbullying (Arabic)								
(Haidar et al., 2017)	2017- Facebook, Twitter	Yes, No	Supervised	Tweet to Senti Strength Feature Vector	SVM	0.93	0.94	0.92
Terrorism (Arabic)								
(Magdy et al., 2016)	2016- Twitter	Pro-ISIS and Anti-ISIS	Supervised	Temporal patterns, Hashtags	SVM	0.87	0.87	0.87
(Kaati et al., 2015)	2015- Twitter	Support or Oppose Jihadism	Semi- Supervised	Data dependent features and data independent features.	AdaBoost	0.56	0.86	0.86

author	year	Platform	DL approache	Features Representation	Agorithme	P	R	F
Religious hate speech (Arabic)								
(Albadi et al., 2018)	2018-Twitter	Hate-non hate	Supervised	Word embedding (AraVec)	GRU-based RNN	0.76	0.78	0.77

Table.II.7. Summary of the Algerian dialect contributions in hate speech detection and their respective results, in the metrics: Precision (P), Recall (R), F1-Score (F).

Hate speech (Algerian dialect)							
author	**platform**	**year**	**Features Representation**	**Algorithm**	**P**	**R**	**F**
(Guellil et al., 2021.)	YouTube	2021	Skip-gram, CBOW	CNN, LSTM, **Bi-LSTM**	0.87	0.87	0.86

Conclusion

The numerous deep learning algorithms for hate speech, as well as the tactics employed by each method, were examined in this chapter. After that, we demonstrated various deep learning research and studies in both English and Arabic. Finally, we spoke about a study that used deep learning models to assess hate speech in Algerian dialect.

Chapter III

Proposed approach for Algerian hate speech detection

Introduction

Considering the negative psychological and societal consequences of increased offensive language usage, whether spoken or written, on paper or on social media, the issue of identifying offensive language and automating its treatment is garnering growing interest from scholars.

In this chapter, we provide a strategy for gathering data from Algerian dialect comments on social media sites such as Facebook, YouTube, and Tik Tok in order to do research on them.

In addition, we provide a potential method based on deep learning, aggregation, and transfer, as well as various phases of initial processing.

1. Global Architecture

We show our global architecture, detailing our basic methods and its fundamental stages, in order to construct an intelligent hate speech detection approach.We begin by gathering information from a variety of social media platforms.

Preprocessing is the second phase we finish, and we present two major preparation perspectives dependent on the models into which our data will be fed. Pretrained language models, such as, have their own preprocessing perspective (Bert-base-arabic, mBert, and mDistilBert), The alternative preprocessing viewpoint is reserved for deep learning models. On the one hand, we feed our preprocessed comments into language models that have already been trained. On the other hand, we turn preprocessed comments into vectors in order to train deep learning. We get these word representations by either utilizing existing pretrained word vectors or training the FastText algorithm on our comments. Figure 2 depicts the main procedures involved in creating an annotated corpus, beginning with comment collecting and filtering, then annotation, and lastly pre-processing.

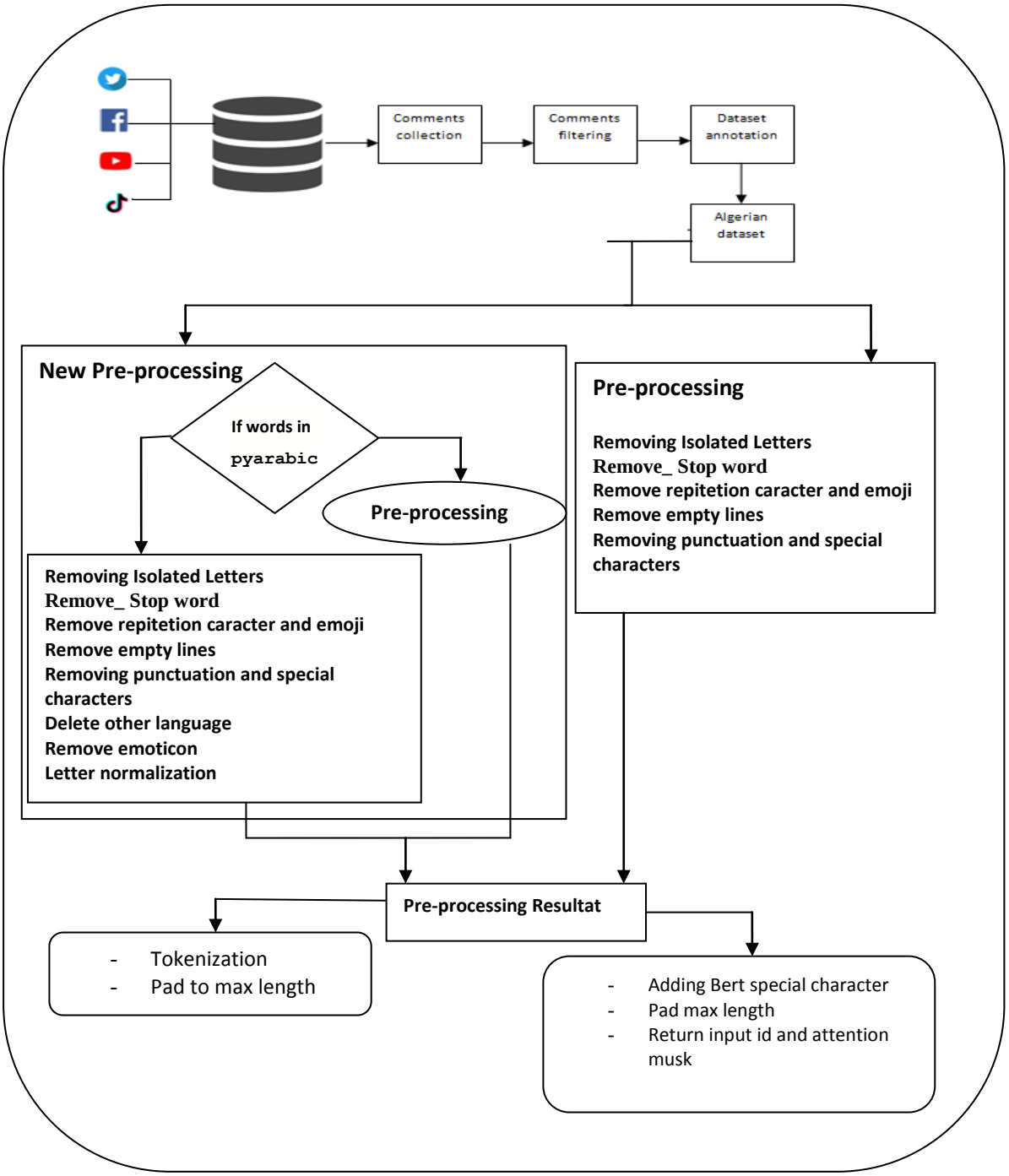


Figure.III.1. General steps for constructing the annotated dataset

2. Methodology detection hate speech

A technique is being developed in order to build an intelligent strategy for detecting cyber hate speech.The dataset must first be collected and prepared. The second step is to convert the text input into characteristics that machine learning algorithms can understand. The third step is to create a deep learning model for smart prediction. Finally, the model's performance in identifying hate data is assessed. The planned technique is summarized in Figure 2



Figure.III.2 An overview flowchart of the methodology.

2.1. Data creation:

We had 14,150 comments collected before from social media sites (this dataset created by Chiker & benkhaoua PFF project) .And we raised it by 10,000, making it 24,150 distinct data in the source from social networking sites (Facebook, Twitter, YouTube, Tik Tok...) and different in a topic where this data varies depending on the issue addressed, such as a politician. social. Art..... It included basic terms from a variety of sectors. We also separated the data into hate speech and non-hate speech.

Figure 3 shows the first step in the data preparation process. It begins with data collection, annotation by two volunteers, cleansing the data of any unnecessary or redundant data, normalization, tokenization, and text vectorization with the goal of features representation.

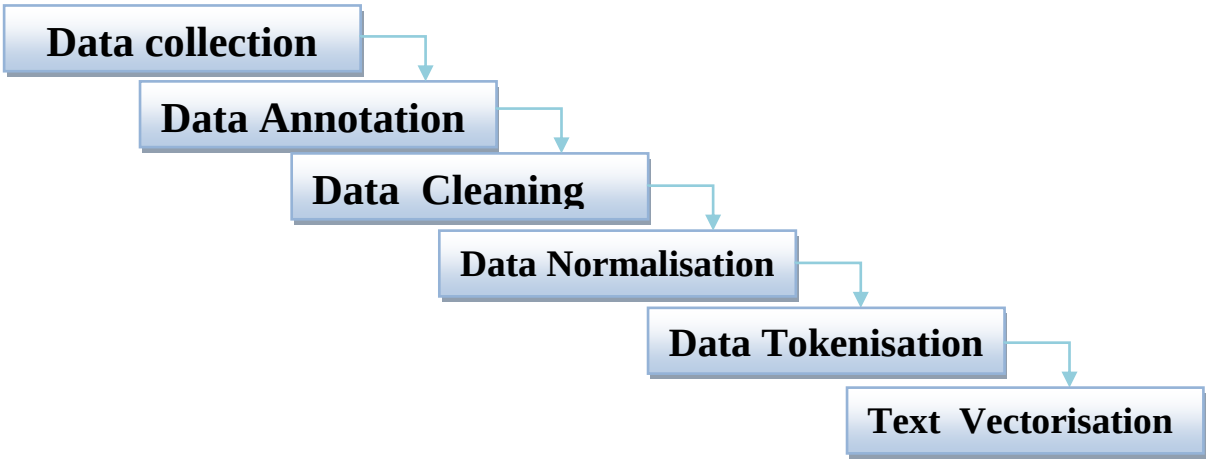


Figure.III.3. The steps of data collection and preprocessing.

2.2. Data collection:

2.2.1. Old data :

The old dataset is made up of comments gathered from three different social media platforms: Facebook, YouTube, and Twitter. They gathered 14,151 comments from Al-Shorouk News Channel, Al-Jazaeryia, Abdelmadjid Tebboune, and other prominent Algerian sites and accounts. Religion, parliamentary politics, Moroccan politics, Misogyny, and other issues of hate speech and cyberbullying, as well as harsh and abusive language, are addressed in these comments.

Table.III.1. Information about old data

Hate speech	
Yes	No
7573	6575

- **Source :**

Table.III.2. Information about Source

Source	Num
Facebook	**5578**
YouTube	**8356**
Twitter	**215**

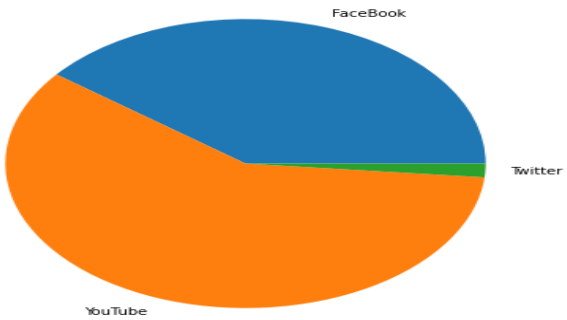


Figure.III.4. Information about Source

- **Topic :**

Table.III.3. Information about Topic

Topic	Num
drugs	854
politics	2039
president's speech	1348
price hikes	847
Morocco politics	1294
religion speech	1537
Misogyny	3187
harraga	293
legislative election	1382
minister Jarad	624
parliamentary politics	744

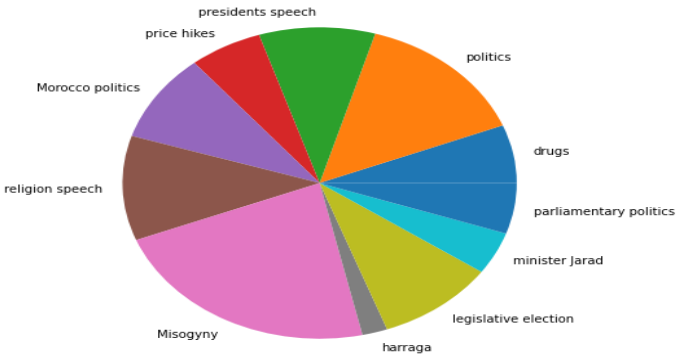


Figure.III.5. Information about Topic

2.2.2. New data :

We increased manually our data collection by adding comments from four other social media platforms: Facebook, Ticktok, YouTube, and Twitter.
And that is by increasing 10,000 comments by searching and collecting various comments by keywords such as Minister Ouyahia, sport, Jamal Belmadi, Numedia and Al-Jazaeryia channels such as Al-Nahar TV, Al-Shorouk, Port TV, various Ramadan programs and Algerian series..... and that is by using keywords in the search engine to show us the most important publications that included that word, for example (the keyword Abdul Majeed Tabun shows us the most important publications and videos, as well as pages dedicated to Abdel Majid Tabun.) , These comments are saved in a xlsx file.

2.3. Comments filtering :

Before being saved in the dataset, all of the obtained comments were filtered as follows:

- ✓ All of the comments posted in languages other were manually erased. Except for Arabic, French and Arabic comments written in character en French.
- ✓ All of a particular tweet's retweets are erased, as are all of the repeated comments.
- ✓ The identical comments with the same contents are filtered to guarantee that each considered comment has a unique text.

We now have a total of 24151 comments after this filtering phase. Figure 06 depicts some dataset details.

- **Source :**
- **Table.III.4.** Information about Source

Source	Num
Facebook	**10240**
YouTube	**12155**
Twitter	**221**
TikTok	**1534**

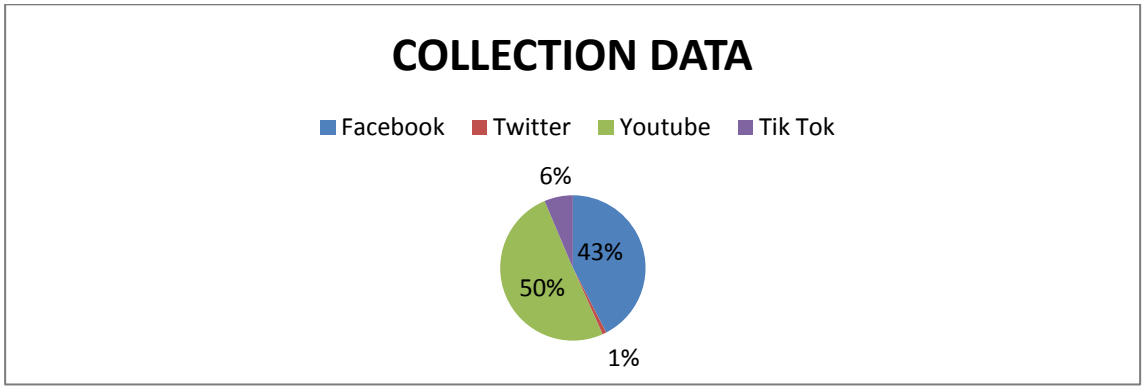


Figure.III.6. Information about the collected dataset

- **Topic :**
- **Table.III.5.** Information about Topic

Topic	Num
drugs	859
politics	2649
president's speech	1484
price hikes	847
Morocco politics	1349
religion speech	1685
Misogyny	5726
harraga	294
legislative election	1382
minister Jarad	624
parliamentary politics	744
TV Shows	2520
humain right	61
art	1129
News	27
Social	2023
Ministre Ouyahia	119
sport	622

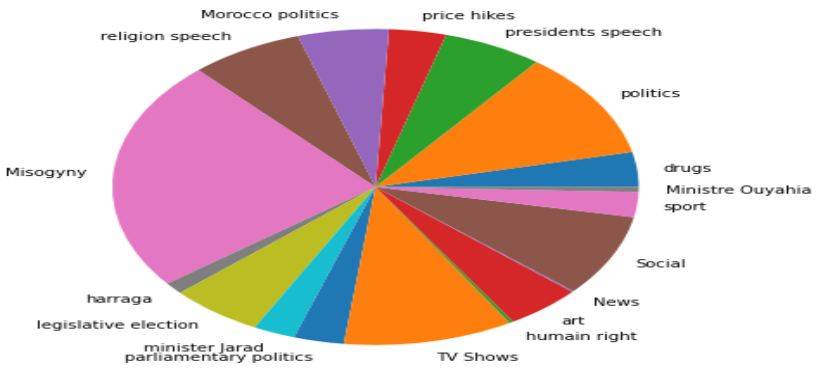


Figure.III.7. Information about the Topic

2.4. Comments annotation

We went on to the manual annotation step after the extraction and filtering phases.

There were three labels in the annotations (hate speech, cyber bullying, and offensive and abusive language). However, after extracting the data, we needed to divide it into two equal portions, as shown in the table 6 : shows some information about the labels.

Final data contains 12,076 comments hate speech and 12,074 non hate speech.

Table.III.6. Information about the dataset

Hate speech	
Yes	No
12076	12074

2.5. Data preprocessing

Text preparation and preprocessing are key components of many NLP activities and applications. That is to account for the informal nature of social media short communications, which include inconsistencies and a heavy usage of slang and colloquial language. Data preparation can also help with disambiguation and the removal of non-indicative sections of phrases in brief tweets, enhancing the accuracy of machine learning models that operate on preprocessed data sets vs raw texts. Table 2 displays an example of a common social media text type from the hate speech dataset.

We find mentions of users, numbers, hashtags, URLs and replace them with the tokens We also find elongated words and convert them into short and standard format; for example, converting (nchallllh to nchalh). All punctuation marks, unknown uni-codes and extra delimiting characters are removed, but we keep all stop words and numbers because our model trains the sequence of words in a text directly. We also convert all data to lower case.

Table.III.7. Example from the dataset on hate speech.

Data	***Classification***	***TOPIC***	***Source***
Voleur harkis il c'est même pas parlé	***Hate***	***Politics***	***Youtube***
J'adore mon bébé d'amour ❤ Libérez Numidia, libérez tout les innocents emprisonnés arbitrairement par une justice injuste Basta l'injustice, Basta le racisme...	***Non hate***	***Misogyny***	***Facebook***
هذه المعزاة الطائشة جزائرية تبقا حاير في هذا الصنف من المساء	***Hate***	***Misogyny***	***Facebook***
دفاع عن النفس فرض واجب على المستعضفين	***Non hate***	***Politics***	***Youtube***

2.6 .Cleaning comments

Comment Cleaning is particularly done as part of data preprocessing to clean the data by filling missing values, smoothing the noisy data, resolving the inconsistency, and removing outliers.

- First, all languages must be deleted except Arabic,French and Algerian dialects.
- All icons have been preserved.
- Replace the duplicate letters with a single instance of the letter
- Keeping empty words and eliminating letters with unknown unicodes
- Letter normalization: unifying the letters that appear in different forms. Example we replace {أ إ ؤ ء آ ئ ى} with {ا}. Also remove all of the Arabic diacritics 'Tash keel' like (tashdid, Fatha, kesra …etc.) and the elongation as (اجـــــــــــراءاتـــــ become اجراءات) ;
- the symbols, punctuation, hashtags, web addresses, diacritics are removed.
- Then we filtered both the Arabic,French and Darija stop words.
- Finally, deleting all the empty lines.

3. Proposed Approach Steps

3.1 . Tokenization

Data tokenization is concerned with dividing the data into set of words based on the white-space delimiter. Herein, the tokenization process is implemented using the Natural Language Toolkit (NLTK) library (Loper, E., & Bird, S., 2002) within Python development framework.

3.2. Padding the sequence to a certain length

Pad sequences guarantes that every sequence in a list is the same length. By default, this is accomplished by padding 0 at the start of each sequence until it is the same length as the longest.

3.3 BERT Tokenization

The built-in tokenizer in the pre-trained BERT model transforms raw input text (as strings of characters) into tokens suitable for the BERT input layer.

Because BERT has its own predefined vocabulary of tokens and a pre-trained embedding vector for each built-in token, it has its own tokenizer one downside of pretrained BERT is that bespoke word embeddings are not possible.

While all of BERT's knowledge and language understanding is limited to the embeddings with which it was pre-trained, it nonetheless performs well with any arbitrary unknown words. The sophisticated technique the BERT uses to handle such out of vocabulary terms is the major cause for this exceptional result. Also, there are domain-specific pretrained models that employ specialized corpora in the area, such as SciBERT for scientific literature and BioBERT for a large corpus of medical language.

To tokenize textual inputs, the BERT tokenizer uses the WordPiece paradigm. Eighty percent of the 30,000 vocabulary terms in BERT are complete words, while the rest are subwords. Out-of-vocab words are broken down into subwords by BERT, and in the case of missing subwords, unknown subwords are broken down further into individual characters until BERT can come up with a plausible approximation of the meaning of the entire unfamiliar word. (Hamdy, 2021)

Special token in Bert

BERT also include some special tokens for its inputs which they are:

1. For sentence-level categorization problems, use the [CLS] token. The token is always placed at the beginning of each phrase.
2. The Next Sentence Prediction task's separator [SEP] token for segment embeddings.
3. The [UNK] token is used to replace vocabulary that does not appear in BERT words or subwords. (Hamdy, 2021)
4. The padding [PAD] token for padding short sentence to make sure that all input sequences in the batch are of the same length when the embedding are stacked together. (Hamdy, 2021)
5. Following WordPieceModel (WPM) tokenization, using BERT tokenizer to tokenize sentences to word pieces.
6. Trimming longer sentences and padding shorter ones to the maximum length so that they are all the same length.
7. Creating attention masks for sentence tokens; this attention mask encoding distinguishes between padded and non-padded tokens such that padded tokens are excluded from attention vector computations.

3.4. Case conversion:

Unifying all text cases such that all characters are lower or upper case is a typical text preparation procedure, with lower case letters being the most common conversion. That manner, repeated terms with various situations would be reduced to just one word, lowering vocabulary size and allowing a machine learning model to better approach our goal.

3.5.Word normalization

Entails stemming, lemmatization, correcting misspellings, and transforming uncommon terms or spelling variations into mainstream terminology. Using a dictionary to map words for conversion is a popular way to dealing with word normalization.

3.6. Punctuation and special character removal:

Non-alphanumeric characters often offer no value to the text understanding process and generate noise, which severely impacts the performance of a machine learning model. As a result, cleaning up text data from these characters can improve model accuracy and performance significantly.

3.7. Transformation of Special Symbols:

This includes substituting Emojis with a representative text that may substantially enrich the context and modify the meaning of the phrase or emphasize the writer's aim. It's also crucial to look at the relationship between a sentence's emotion and unique symbols like emojis.

3.8. Segmentation:

Hashtags are well-known in social media content, and one prominent example where segmentation is required is to separate concatenated hashtags into independent words, thereby improving the meaning of the phrase.

3.9. Normalization of user mentions and links:

This includes deleting duplicate user mentions and links. Because most pretrained models have a word embedding, substituting link with the string "HTTP" is another frequent link processing method.

3.10. Creating sentence token attention masks:

This attention mask encoding distinguishes between padded and non-padded tokens, allowing padded tokens to be excluded from attention vector computations.

3.11 . Grouping :

A collection of words that have the same meaning and are derived from a single word. Two or more words with the same meaning can be spelled differently, for example, by increasing the number of consecutive repeated letters.

This is called as grouping, when we return these words to their original form.

normalization of speech effects such as reducing all redundant letters in a word to a standardized form e.g the elongated form of the word

(بوتتتتتفليفة'' , بوتوتوفليقة'')=> '' بوتفليقة''.

3.12 . Transcripted :

It means to write Arabic words in Latin, for example,as a sentence:

ان شاء الله=> inshalah

4. Features extraction

The next stage in an NLP application pipeline is to extract relevant representations of the text in numerical format that are acceptable for predictive machine learning models after getting a cleaned and normalized version of the textual dataset. The basic purpose of text feature extraction is to keep contextual and semantic linkages as near as feasible. A document's representation vector can be an average of the vector representations of the words in the document, or it can be a learnt representation for the entire document/sentence using a representation approach like word2Vec.

We use three different algorithms for feature extraction which are, Word2vec (Mikolov, 2013)FastText (Joulin, 2016)and TF-IDF ,We use Word2vec and TF-IDF with classic methods [machine learning] and we use FastText with Deep learning methods. Both Word2vec and FastText describe two architectures for computing continuous vector representations, the Skip-Gram (SG) and Continuous Bag-Of-Words (CBOW).

4. 1. Word2Vec:

W2V (Word2Vec) is a neural network-based approach for learning high-quality continuous vector representations from a vast corpus of text with billions of words. The semantic similarity between words is preserved when using a W2V model to represent words. W2V assigns each word a representation that reflects its meaning and connections with other words. The curse of dimentionality problem that was present in one hot encoding was successfully mitigated with W2V. To learn the underlying representations for each word, it employs either a Continuous Bag of Words or a Skip-gram neural network architecture. Both designs are built on a fictitious job in which each pair of words in the corpus is selected to educate the model that they are co-occurrences. Each word representation is a vector of the same length as

the hidden layer of the W2V neural network, and the training weights are considered the final representation.

4.2. TFIDF

TFIDF is a statistical approach for determining the importance of a word in a corpus based on the word frequency relative to the number of documents to account for the fact that some words appear more frequently in general.

4. 3.Fasttext :

FastText is a lightweight, open-source framework for learning text representations and classifiers. It may be used with any type of computer. Models can be scaled down to fit on mobile devices in the future.

5. Classification :

5.1. Machine learning

We employ TF-IDF and Word2Vec with five classification algorithms (RandomForset (RF), multinomial naive bayes, LogisticRegression, SGDClassifier, and LinearSVC) for the traditional technique. We were motivated by the categorization algorithm presented by for their implementation phase (Joshi et al.,2016).

Support Vector Machine, Logistic Regression, and Nave Bayes Text classification frequently employs these models. With the premise that the characteristics do not interact, Nave Bayes models label probabilities directly. SVMs and Logistic Regression are linear classifiers that predict classes based on a combination of feature ratings. (Pedregosa F, 2011)

5.2. Deep learning :

For the deep learning classification, we use the following models CNN, LSTM,RNN Bi-LSTM, GRU. For each model, we use different layers.

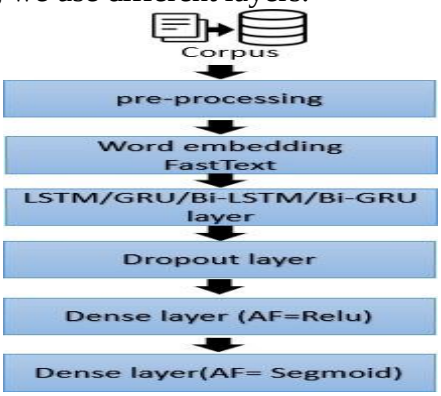


Figure.III.8. DL architecture **.**

RNN :

Starting with the Recurrent Neural Networks (RNNs) fundamental architecture, which served as the foundation for processing a sequence of textual data for various NLP applications.

RNNs use a variety of architectures, such as many-to-many RNNs for Named Entity Recognition and machine translation jobs, one-to-many RNNs for music production, and many-to-one RNNs for sentiment analysis, to name a few.

LSTM:

The LSTM RNNs were created by the authors at (Hasim Sak, 2014)to tackle the vanilla RNNs' long-term dependence problem while working with lengthy text sequences, namely because of the vanishing gradients problem. A memory cell, an input gate, an output gate, and a forget gate make up a typical LSTM unit. The three gates control the flow of information into and out of the cell, and the cell remembers values across arbitrary time periods. The objective of LSTM network is to solve the problem of the vanishing gradient of neural networks (Gers, 20020.)In addition, LSTM is very efficient for handling sequences of data, or sequences of words in case of textual data .

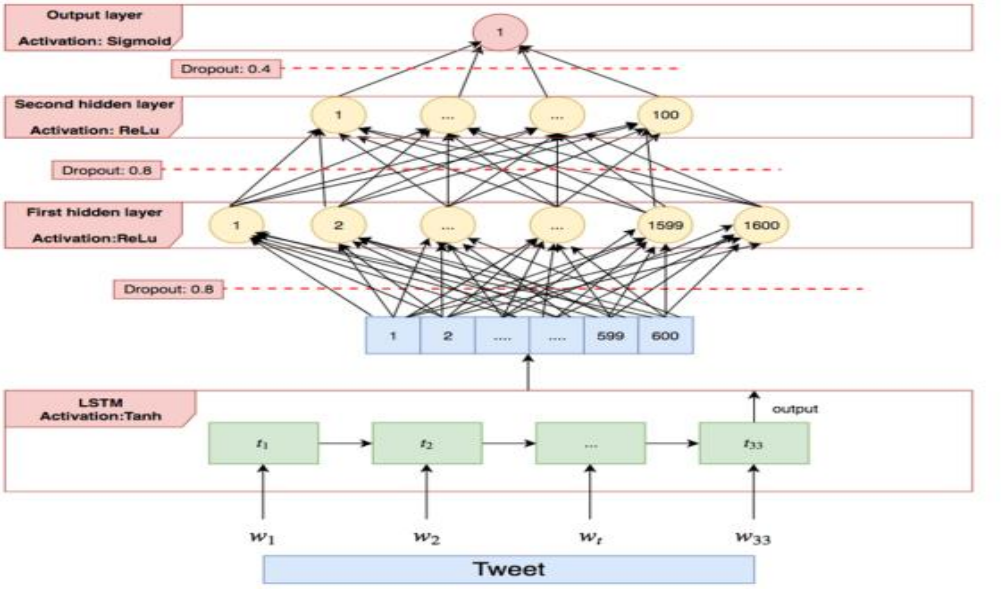


Figure.III.9. LSTM architecture **.**

CNN:

The first layer of CNN is a randomly initialized word embedding layer, which converts words in sentences into feature maps. The feature map is then scanned by the CNN layer. The output layer is subjected to global maxpooling. The scores are then fed into a single feed-forward (fully-connected) layer, whose output is sent via a sigmoid layer that predicts the output classes.

5.3 Transfer learning

Bert Model:

For each input token embedding, the BERT model processes all other token embeddings to determine the word's context and provide a superior embedding for that token. Because each word is independent of the others, this method can handle the processing of all tokens in simultaneously. This also allows BERT to accept input sentences of varying durations, whereas a recurrent neural network would manage this by sequentially processing each work one after the other. The process of taking a set of token embedding and producing better ones

are repeated 12 times across the 12 layers of BERT.The following figure 10 shows shows the 12 layers of BERT architecture and how the input for each layer:

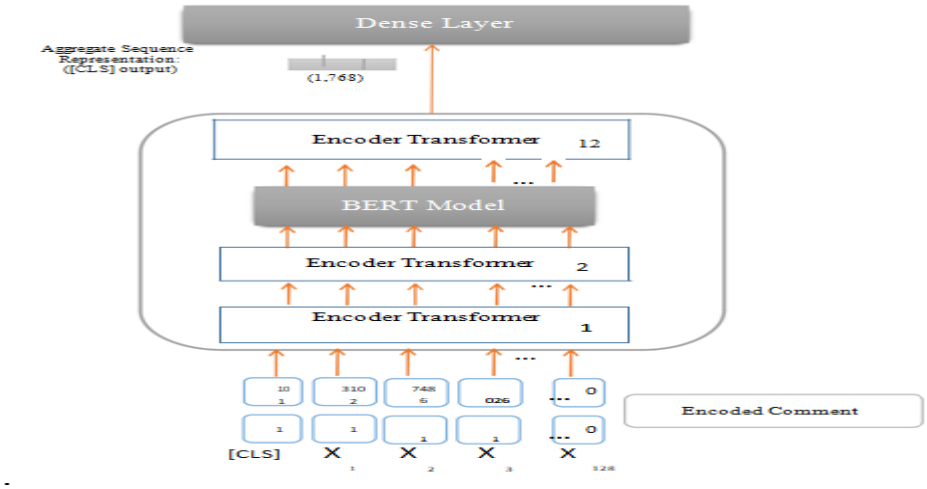


Figure.III.10 . BERT Model Representation.

Multilingual BERT (mBERT):
Multilingual BERT (mBERT (Jacob Devlin, 2019))is gaining popularity as a contextual representation for various multilingual tasks, such as dependency parsing (Straka, 2019), cross-lingual natural language inference (XNLI) or named-entity recognition (NER) (Telmo Pires, Eva Schlinger, and Dan Garrette., 2019)(Shijie Wu and Mark Dredze, 2019)

Distil Bert Multilingual
Description The model is based on a concatenation of Wikipedia articles in 104 languages. The model consists of 6 layers, 768 dimensions, and 12 heads, with a total of 134 million parameters (compared to 177M parameters for mBERT-base). DistilmBERT is twice as quick as mBERT-base on average.The three transformer models listed above have only been educated in current standard Arabic. As a result, they may find it challenging to communicate in Algerian dialects. As a result, we used our corpus to undertake further pretraining (or fine-tuning) for the AraBERT, mBERT, and distilmBERT models. This is done in order to familiarize the transformer language models with its vocabulary.

BERT + Pooled Stacked Bi-LSTMs
We added Bi-LSTMs to the BERT model and used the tokens' final BERT outputs as their inputs.Stack architecture beats a basic one, according to previous studies (Chakrabarty, 2019). As a result, instead of employing only one layer, we use two layered BiLSTMs.

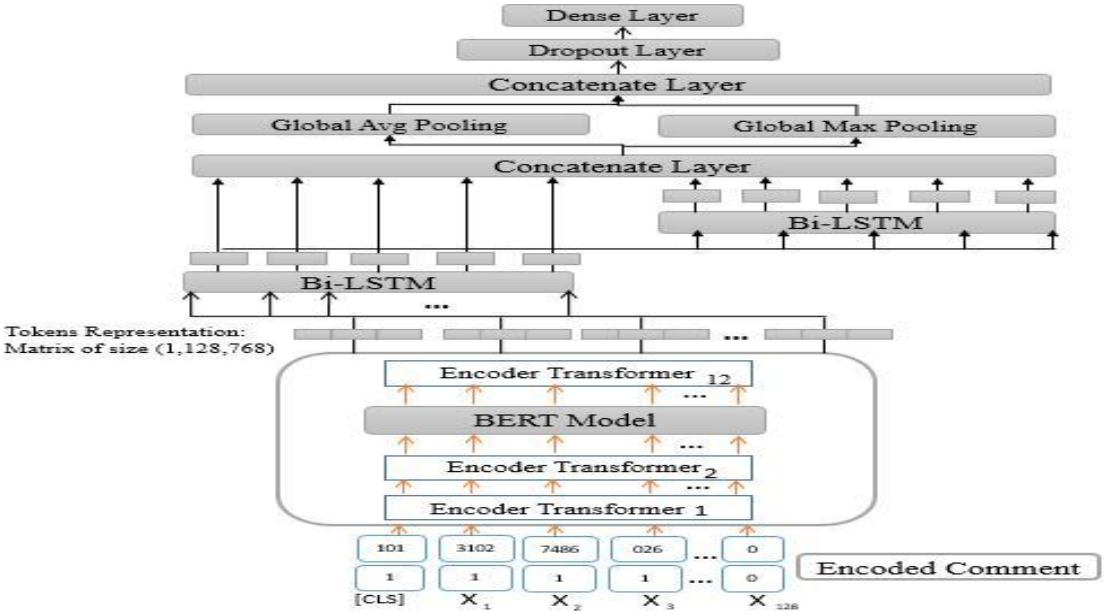


Figure.III.11. BERT + Pooled Stacked Bi-LSTMs Model.

BERT + Pooled stacked Bi-GRUs

In the third model, We implement the identical model (BERT+BiLSTM) but with Bi-GRU layers instead of Bi-LSTM layers to see which RNN variation performs better with BERT model for the task of offensiveness detection.The use of GRU units and LSTM ones was evaluated ina an identical architectures in (Saad. M. K, 2010), and the results have shown that the architecture based on GRUs outperformed the one based on LSTMs. Furthermore, it produces the best results of all the provided models.Figure 12 shows the illustration of BERT + Pooled Stacked Bi-GRUs model.

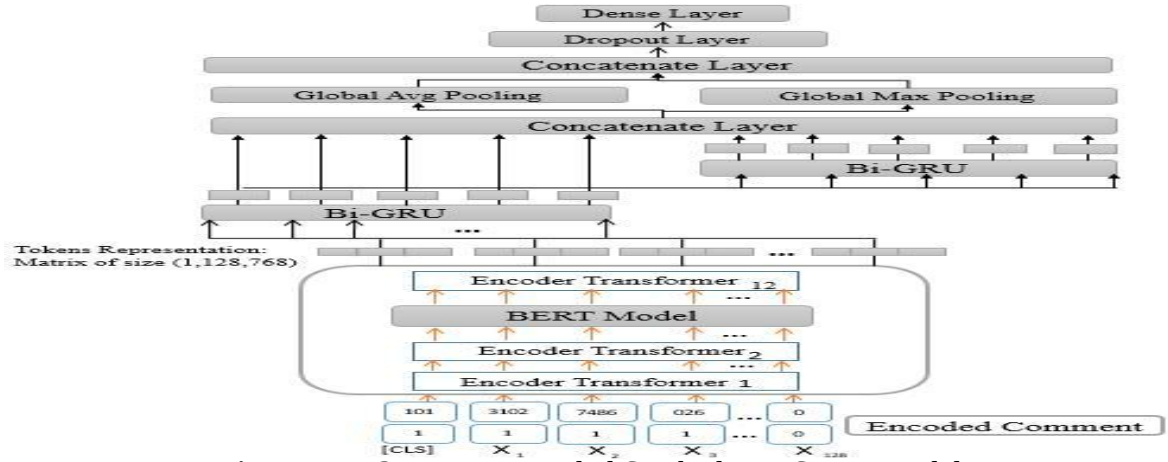


Figure.III.12. BERT + Pooled Stacked Bi-LSTMs Model.

6. Evaluation Measures

Confusion Matrix :

If the data set has a skewed distribution of observations by target class, accuracy is not the preferred performance measure and a much more informative method is to investigate the confusion matrix (Géron, 2017)

It is based on a set of predictions for observations where the actual values of the outcome variable are known. Therefore, the actual and the predicted class labels can be compared. Figure 16 shows the structure of a confusion matrix.

The actual classes are represented as rows and the predicted classes as columns. So in case of a binary classification the matrix has two rows and two columns. In each of the four quadrants, the numbers of observations are placed according to their actual and predicted labels. The cases where the classifier was right and the actual class matches the predicted class can be either true negatives (TN) or true positives (TP). A classifier that makes no mistakes would have only TPs and TNs, i.e. non-zero values the main diagonal (top left to bottom right). The errors, however, can be false negatives (FN) or false positives (FP).



Figure.III.13. Confusion Matrix**.**

The used performance evaluation measures are accuracy, precision, recall, and F1 measure. The accuracy is the ratio of correctly classified Hate and Normal tweets over all the correct and the incorrect number of classified data Where the accuracy is formulated in Equation 1

Accuracy measures the overall performance of the classifier.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \text{ ----------(1)}$$

Precision and Recall

Precision : and recall can be derived from the confusion matrix. As defined in Equation 2, precision is calculated as the ratio of the number of true positives divided by the sum of the true positives plus false positives. It describes how well a model can predict the positive class.

Recall : on the other hand, is the ratio of the number of true positives divided by the sum of the true positives and the false negatives, see Equation 3 It describes to what extent a model identifies instances of the positive class.

$$precision = \frac{\text{TP}}{\text{TP+FP}} \text{ ----------(2)}$$

$$recall = \frac{TP}{TP+FN} \text{ ----------(3)}$$

F1 Score : The F1 score is even more concise, as it combines precision and recall into a single
metric. It is their harmonic mean, as defined in Equation 4 In contrast to the regular mean, which treats all values equally, the harmonic mean gives far more weight to low values. Consequently, the classifier only gets a high F1 score if both precision and recall are high
F1 score if both precision and recall are high (Géron, 2017).

$$\textbf{F1 Score} = \frac{2*precision * Recall}{Precision + Recall} \text{ ----------(4)}$$

Conclusion :

In this third chapter, we introduce the many components we complete for Algerian dialect. The preprocessing step begins with two types of preprocessing, depending on the model into which the preprocessed comment will be input. Then, for the Algerian dialect, providing the major proposed ways to hate speech identification, which include machine learning classifiers, deep learning classifiers, and transfer learning classifiers. Creating a deep learning architecture and fine-tuning pre-trained language models.

Chapter IV :

Test and Evaluation

Chapter IV: Test and Evaluation

Introduction :

The experimenting of our proposed approach will be presented in this chapter. To begin, we discussed the environment as well as the many modules we utilized to train, fine-tune, and test our techniques. Then we went over the various tools and libraries we utilized, as well as the actions we took during our experiment. Then, to assist us understand our work better, we displayed the data we obtained in various forms, such as tables and graphs, and we contrasted these results, which are represented by the f1-score. Finally, we evaluated our effort by interpreting and discussing our findings.

1.Environment and Development Tools :

1.1. Google Colaboratory[3] :

Google Research created Collaboratory, which is frequently abbreviated as "Colab." Colab allows anyone to write and run Python code directly from their web browser. It's a cloud-based platform for directly training machine learning models, and it's especially well-suited to machine learning, data analysis, and education. We don't need to install anything on our PC except a browser. To put it another way, Colab is a hosted Jupyter notebook service that requires no installation and gives you free access to Google processing resources like as GPUs and TPUs.

Session Duration :

One Google colab session provides 12 hours of continuous execution.

RAM Space Available :

The amount of RAM accessible is 12 GB, and there is no way to increase it.

Available storage space :

In addition to Google Drive storage space, the Google colab environment allows you to use more than 30 GB of storage space for storage during a session, which means that all python modules and files saved on this drive will be deleted once the session is completed.

Types of GPUs :

The GPUs that Colab supports may change in the future. This fluctuation is necessary in order to maintain Colab's materials free. Colab regularly uses Nvidia's K80, T4, P4, and P100 GPUs. It is not possible to choose the GPU Colab makes available at any given time. (n.d., research.google.com)

1.2. Data extraction:

1.2.1.Python :

Python is a high-level programming language that may be used for a variety of tasks. Python stresses code readability, as seen by its heavy use of indentation. Its language components and object-oriented approach are designed to help programmers write concise, logical code for small and large-scale applications.

Python is a dynamically typed, garbage-collected language. It can handle a wide range of programming paradigms, including structured (particularly procedural) programming, object-

[3] Bienvenue dans Colaboratory - Colaboratory (google.com)

oriented programming, and functional programming. Python is frequently referred to as a "batteries included" language because of its large standard library.[4]
We utilize Python 3.7.10 and the following modules in the Google colab execution environment

1.3. Implementation of Deep Learning

1.3.1Keras:

Keras is a high-level neural network API written in Python that can be used with TensorFlow, CNTK, and Theano. Keras is the way to go if you need a deep learning package that allows for easy and speedy experimentation (through user-friendliness, modularity, and extensibility). Convolutional and recurrent networks are supported, as well as hybrids of the two. Runs flawlessly on both CPU and GPU.
Keras is utilized to create deep learning models in our research.

1.3.2Genism :

Gensim is a free open-source Python toolkit for encoding documents as
semantic vectors. It was created to analyse unstructured digital texts ("plain text") using unsupervised machine learning methods like Word2Vec and FastText.

1.4. Pre-processing and preparation of textual data :

1.4.1.Pandas[5] :

Pandas is an open-source, BSD-licensed library for the Python programming language that provides high-performance, easy-to-use data structures and data analysis tools for textual data pre-processing and preparation.
Pandas is a NumFOCUS-sponsored project. This will help pandas become a world-class open-source project by allowing donors to donate to the project.
To read datasets, we used Pandas.

1.4.2.Numpy[6] :

Numpy is a Python module with open source code that is released under a liberal BSD license. Numpy includes a number of mathematical functions, including random number generators, linear algebra techniques, and Fourier transforms. It also has a high-level syntax and is compatible with a variety of hardware and processing systems, as well as distributed, GPU, cloud computing, and sparse array libraries.

1.4.3.Re :

This module was added in Python 1.5 and provides Perl-style regular expression patterns. The regex module was provided, which allowed Emacs-style patterns in previous versions of Python. The regex module was completely removed in Python 2.5.

[4] Welcome to Python.org

[5] pandas - Python Data Analysis Library (pydata.org)

[6] NumPy

The Re module is used in our work as a pre-processing step to build regular expressions to find hashtags in texts and to replace URLs, emoticons, user remarks, and integers with regular expressions.

1.4.4.Regular expressions :

Regular expressions (also known as REs, regexes, or regex patterns) are a small, highly specialized computer language that is integrated in Python and accessible via the re module21. Punctuation, emoticons, URLs, redundancies, and words with less than two characters were all removed with it.

1.4.5.NLTK[7] :

NLTK is a popular Python programming language for working with human language data. It includes a suite of word processing libraries for classification, encoding, derivation, tags, analysis, and semantic reasoning, wrappers for industry-strength NLP libraries, and an active discussion forum, as well as easy-to-use interfaces to more than 50 lexical collections and resources, such as WordNet. For Arabic, French, and Algerian languages, we used the NLTK list of stop words.

1.5 For BERT Models Exploitation

1.5.1.Transformers:

Transformers (formerly pytorch-transformers and pytorch-pretrained- bert) provides general-purpose architectures for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between Jax, PyTorch, and TensorFlow.

1.5.2.Tensorflow :

Deep learning is a type of machine learning that uses algorithms inspired by the structure and function of the brain.

Google's second machine learning framework, TensorFlow, is used to design, develop, and train deep learning models. The TensorFlow library may be used to do numerical computations. These computations are carried out using data flow graphs, which may not appear to be very interesting in and of themselves. These networks' nodes represent mathematical processes, while the edges represent the data that is transported across these linkages, which are typically multidimensional data arrays or tensors. The library runs on a wide range of CPUs and GPUs and can be used on a variety of platforms, including mobile.

For our hands-on work in the Google colab environment, we used the most recent version of Tensorflow.

1.5.2.BertTokenizer :

Bert comes equipped with his own tokenizer, BertTokenizer. It is built on WordPiece tokenization, which allows it to handle a broader range of Out of Vocabulary (OOV) words. We employ it in our method to finetuning the pretrained Bert model with the multilingual parameter in order to improve tokenization performance on our multilingual dataset.

[7] Natural Language Toolkit — NLTK 3.6.2 documentation

1.6. For Performance Evaluation Metrics

1.6.1.Sklearn.metrics :

Sklearn.metrics is a scikit-learn module that provides a number of metrics for assessing the performance of machine and deep learning models. This library was chosen because it has functions for calculating roc-auc, recall, precision, and f1 scores. We can also print confusion matrixes and categorization reports using this module.

1.7. For grouping

1.7.1.Soundex :

Soundex is a phonetic algorithm that assigns values to words or names in order to compare pronounciation similarity. I'll build a Python implementation for this post.

It doesn't take much reflection to see that the entire topic of phonetic algorithms is a minefield, and Soundex's utility is quite limited. In reality, after building this implementation, I've come to the opinion that it's quite mediocre, but at the very least, coding it gives you a better grasp of how it works, and thus its utility and limitations.[1]

1.8. For transcript letters :

Unicode :

Is a specification that intends to list and code every character used in human languages.[2]

New languages and symbols are added to the Unicode specifications on a regular basis.

2. Data Used for Training and Test :

We discussed the data in great detail, including how it was acquired, sorted, and processed.... Aside from the components and types of data included in both new and old data... In Chapter 3, we mentioned them.

We left one column for hate speech after removing two columns on cyberbullying and offensive language from the old data. We have taken care of hate speech and its connotations, as well as the largest collection of abusive phrases on social media sites, because hate speech covers cyberbullying and unpleasant language.

Comments were gathered and manually classified according to source, topic, and whether or not they contained hate speech.

Hate speech and non-hate speech statistics were equilibrated from the comments (12076 hate and 12074 non hate

[1] https://www.codedrome.com/the-soundex-algorithm-in-python

[2] https://www.unicode.org

	Id	comments	Topic	hate	Source
0	1	اعلام ضد الجزائر و ضد الشعب الجزائري.	drugs	no	FaceBook
1	2	تمثيل و سيناريو رديء جدا و عنوان خادع	drugs	yes	FaceBook
2	3	اقعد صحيح \n خصني قهوة بشيح لا طيح □□□□	drugs	no	FaceBook
3	4	دفاع عن النفس فرض واجب على المستعضفين	politics	no	YouTube
4	5	فالاخير طفرت غير فالشعب الماضي انتهاء	politics	no	YouTube

Figure.IV.1. Exempl Dataset

2.1.Experimentation Steps :

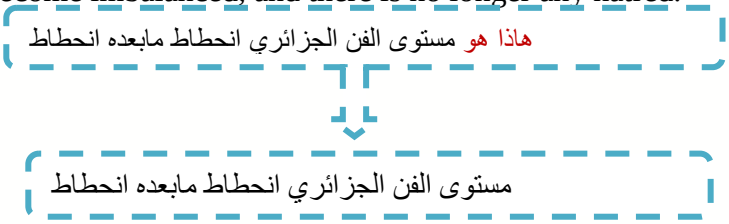
2.1.1.For Pre-processing :

Deleting Isolated Letters and removing special characters are part of the preprocessing of comments before to training deep learning models. After each phase of preprocessing, the following diagrams provide an example of comment transformation. After each preprocessing step, we show the result of a specific remark sample.

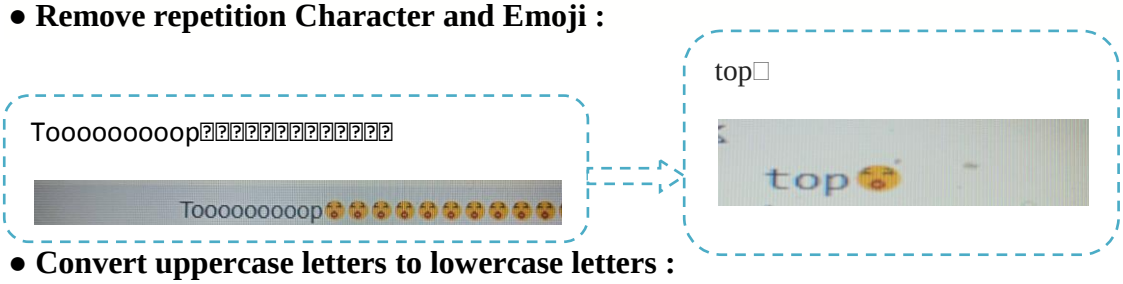
- **Remove_ Stop word :**

Algerian dialect is a difficult dialect to comprehend since it is devoid of norms and linguistic regulations. There are also Arabic-written French words and even numerals. It also lacks any words that have been stop words .

Stop words are words that are often filtered out before a natural language is processed. These are the most frequent words in any language (articles, prepositions, pronouns, conjunctions, and so on), and they don't add anything to the text. However, because it may have a connotation that incites hostility in Algerian dialect, it cannot be eliminated. If we take it out, the phrases become imbalanced, and there is no longer any hatred.



- **Remove repetition Character and Emoji :**

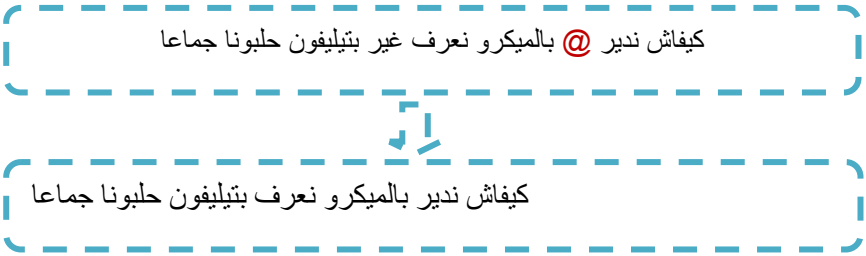


- **Convert uppercase letters to lowercase letters :**



- **Remove empty lines .**

- **Removing punctuation and special characters :**



- **Don't remove the emoji :**

Emoji apparently views as potentially indicating hate speech include a rainbow flag, various animals, and a fist. Emoji that could be related to "sexualized text" include a tongue, an eggplant, a peach, and a hot dog.
As a result, we left all emojis, regardless of whether they reflect hate speech or not.

- Comment **non hate speech**

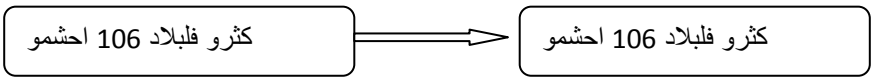


Comment **hate speech**



- **Don't removing numbers :**

We didn't remove the numbers since they have a hate speech connotation in Algerian dialect. Algerians utilize numerals to convey abusive remarks in their dialect.



- **Don't use the Arabic dictionary :**

li fi nos tchbhlk mn3rf fah

Bonjour, j' aimerais bien la vostfr s' il vous plait.
Je ne demande qu' à partager mais je préfère comprendre.

● **Don't use the ISRIStemmer :**

Algerian dialects are not subject to Arabic regulations and laws, and they lack unique terminology. When we employ the terms ISRIStemmer, they lose their meaning and become incorrect.

● **Do nott use prefixe and suffixe :**

```
CUSTOM_PREFIX_LIST = [u'',u'كال', u'أفبال', u'أفك', u'فك', u'
أولل', u'ول', u'أوال', u'ولل', u'فب',u'ألل', u'لل',u'وكال', u'
أوب', u'بال', u'أكال', u'ال', u'وب', u'أوبال',u'وبال', u'فكال',
  u'فلل', u'وك', u'أل', u'فال', u'وال', u'أوكال', u'أفلل', u'
أفل', u'فل', u'أال', u'أفكال',  u'أبال', u'أفال', u'أفب', u'
فبال',u'ال']
```

بهلولة انتاع الصح → بهلل نتع الصح

2.1.2. For Grouping :

We used it in Latin letters only

Exempl Comments :

'les ataques des hakers marocains et des medias marocains sont de loin pire que ce que echorouk avait difuser seulement quand ca vient de algerie cela vous fait chier

'le peuple algeriens chaine chourouk tv contre propagande'

'yakdeb 3lik hna farhanin chorouk chkoun galak halat istinfar'

	word	most_occ
0	algeriens	Algerie
1	chorouk	Chourou

Resultat groping :

'les ataques des hakers marocains et des medias marocains sont de loin pire que ce que echourouk avait difuser seulement quand ca vient de algerie cela vous fait chier sur pantalons cela ne merite pas tout ce tapage il faut demander au maroc et au makhzen areter les ataques contre alerie',

'le peuple algerie chaine chourouk tv contre propagande',

'yakdeb 3lik hna farhanin chourouk chkoun galak halat istinfar'

2.1.3. For Trnscript letters :

```
buckArab = {
"a":"أ","e":"ا","e":"ء","i":"إ","jo":"جو","ro":"رو","pa":"پا","po":"پو","pi":"پي","ta":"طا","to":"طو","
ti":"طي","kha":"خا","kho":"خو","khi":"خي","gha":"غا","gho":"غو","ghi":"غي","7a":"حا","7o":"حو",
"7i":"حي","ba":"با","bo":"بو","bi":"بي","sa":"سا","so":"سو","si":"سي","da"
:"دا","do":"دو","dh":"ذ","di":"دي","fa":"فا","fo":"فو","fi":"في","ja":"جا","ji":"جي","ha":"ها","ho":"هو
","hi":"هي","ka":"كا","ko":"كو","ki":"كي","la":"لا","lo":"لو","li":"لي","ma":"ما","mo":"مو","mi":"مي"
,"na":"نا","no":"نو","ni":"ني","ra":"را","ro":"رو","ri":"ري","ta":"تا","to":"تو","ti":"تي","wa":"وا","w
o":"وو","wi":"وي","ya":"يا","yo":"يو","yi":"يي","za":"زا","zo":"زو","zi":"زي","dha":"ظا","dho":"ظو
","dhi":"ظي","cha":"شا","cho":"شو","chi":"شي","g":"ڤ","dha":"ذا","dho":"ذو","dhi":"ذي","tha":"ثا"
,"tho":"ثو","thi":"ثي","ga":"ڤا","go":"ڤو","gi":"ڤي", "bo":"بو", "a":"أ","p":"پ",  "9a":"قا"  ,"9o":"قو"
,"9i":"قي","3a":"عا","3o":"عو","dhed":"ضد" ,"3i":"عي",
"aaa":"ة","dha":"ضا","dho":"ضو","dhi":"ضي","g":"ڤ","ch":"ش", "b":"ب", "t":"ت", "th":"ث",
"j":"ج","7": "ح", "kh":"خ", "d":"د", "dh":"ذ", "r":"ر", "z":"ز", "s":"س", "ch":"ش", "S":"ص",
"dhe":"ض", "T":"ط","3":"ع"  ,"dh":"ظ", "gh":"غ",  "f":"ف", "a":"ى",  "9":"ق", "k":"ك", "l":"ل",
```

Commentair : وقت بوتفليقا كنا عيشين حالا ⟹ w9t botfli9a kna 3ichin 7ala

2.1.4. For Pre-processing Bert models :

- **Tokenization**

" لازم تاخدي دواء لاسترخاء الاعصاب " → [' لازم ', ' تاخدي ', ' دواء ', 'لاسترخاء ', ' الاعصاب ']

- **Special tokens adding :**

To express the beginning and end of a sentence, specific tokens [SEP] and [CLS] must be included in each sentence of the dataset.

لازم تاخدي دواء لاسترخاء الاعصاب ⟹ “[CLS] لازم تاخدي دواء لاسترخاء الاعصاب [SEP]”

- **Input IDs array:**

```
[    2 12634 16571 ...     0     0     0]
```

- **Attention mask array :** `[1 1 1 ... 0 0 0]`

2.1.5. New Pre-processing :

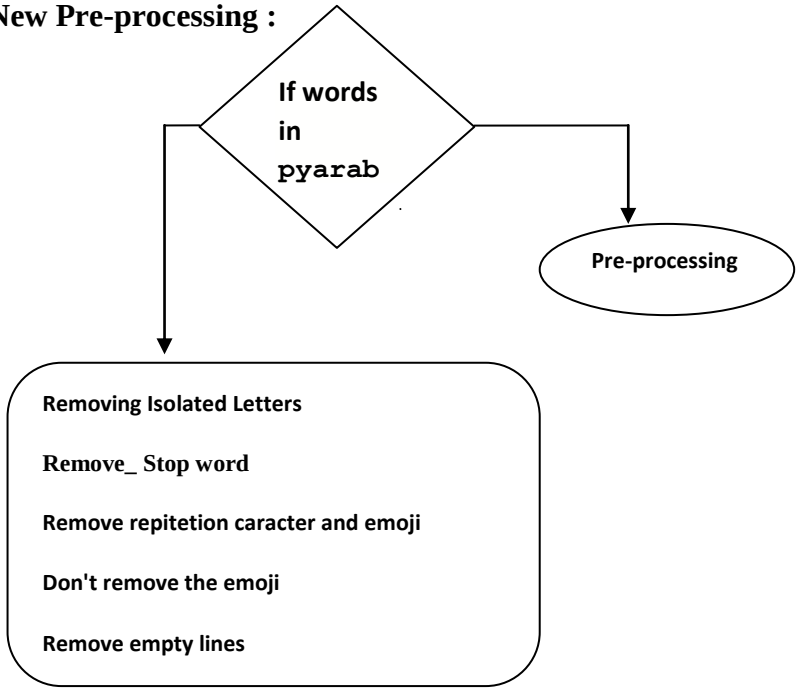


Figure.IV.2. New Pre-processing

3.Model Implementation

3.1.Dataset Preparation:

I.Data Collection 24150

A1= **Data24kArabic**=data24k=> **Pre-processing**=> grouping=>applique models=> Test and evaluation.
A2= **Data24kArabic**=data24k=>**New Pre-processing**=> grouping=> applique models=> Test and evaluation.
B1=**Data24KarabicFR**=data24k=>**Pre-processing**=>transcript=>grouping=>applique model=> Test and evaluation.
B2=**Data24KarabicFR**=data24k=>**NewPre-processing**=>transcript=>grouping=>applique model=> Test and evaluation.
C1=**Data48karabicFR**=data24k +data24k transcript without processing=48k =>**Pre-processing**=>grouping=>applique model=>Test and evaluation.
C2=**Data48karabicFR**=data24k+data24k transcript without processing=48k=>**New Pre-processing**=>grouping=>applique model=> Test and evaluation.

II.Old Data 14150

D=Old data For colleagues 14K=>applique model=>Test and evaluation.

III.Data Creation 10000

E1=Data10kArabic=data10k=>**Pre-processing**=>grouping=>applique model=>Test and evaluation.
E2=Data10kArabic=data10k=>**New Pre-processing**=> grouping=>applique model=> Test and evaluation.
F1=Data10KArabicFR=data10k=>**Pre-processing**=>transcript=>grouping=>applique model=> Test and evaluation.
F2=Data10KArabicFR=data10k=>**NewPreprocessing**=>transcript=>grouping=> applique model=> Test and evaluation.
G1=Data20karabicFR=data10k+data10k transcript without processing=10k => **Pre-processing**=>grouping=>applique model=>Test and evaluation.
G2=Data20karabicFR=data10k+data10k transcript without processing=10k => **New Pre-processing**=>groping=>applique model=> Test and evaluation.

3.2.Models :

The following tabel explains thedata and the models

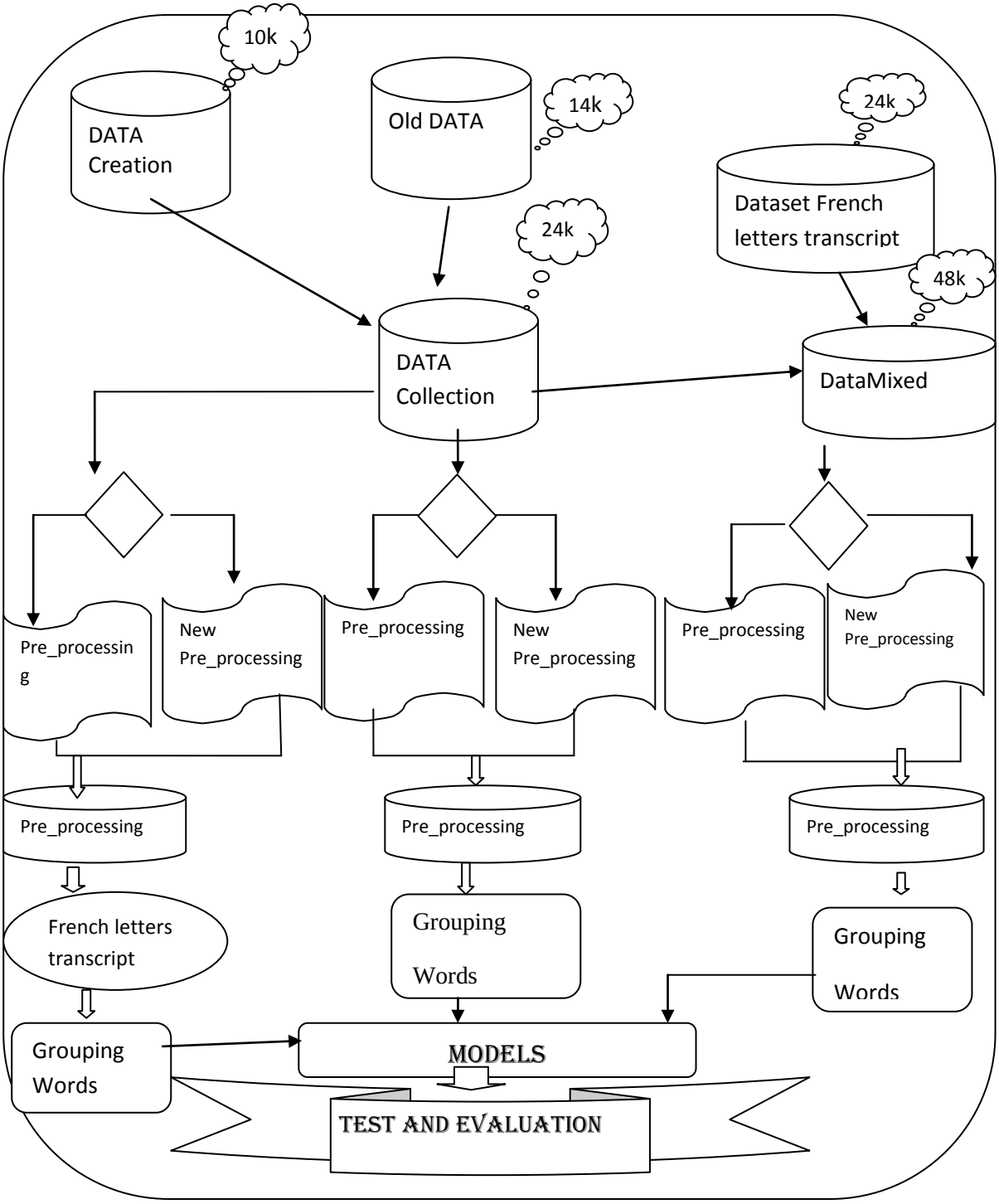


Figure.IV.3. General steps for constructing the annotated dataset 1

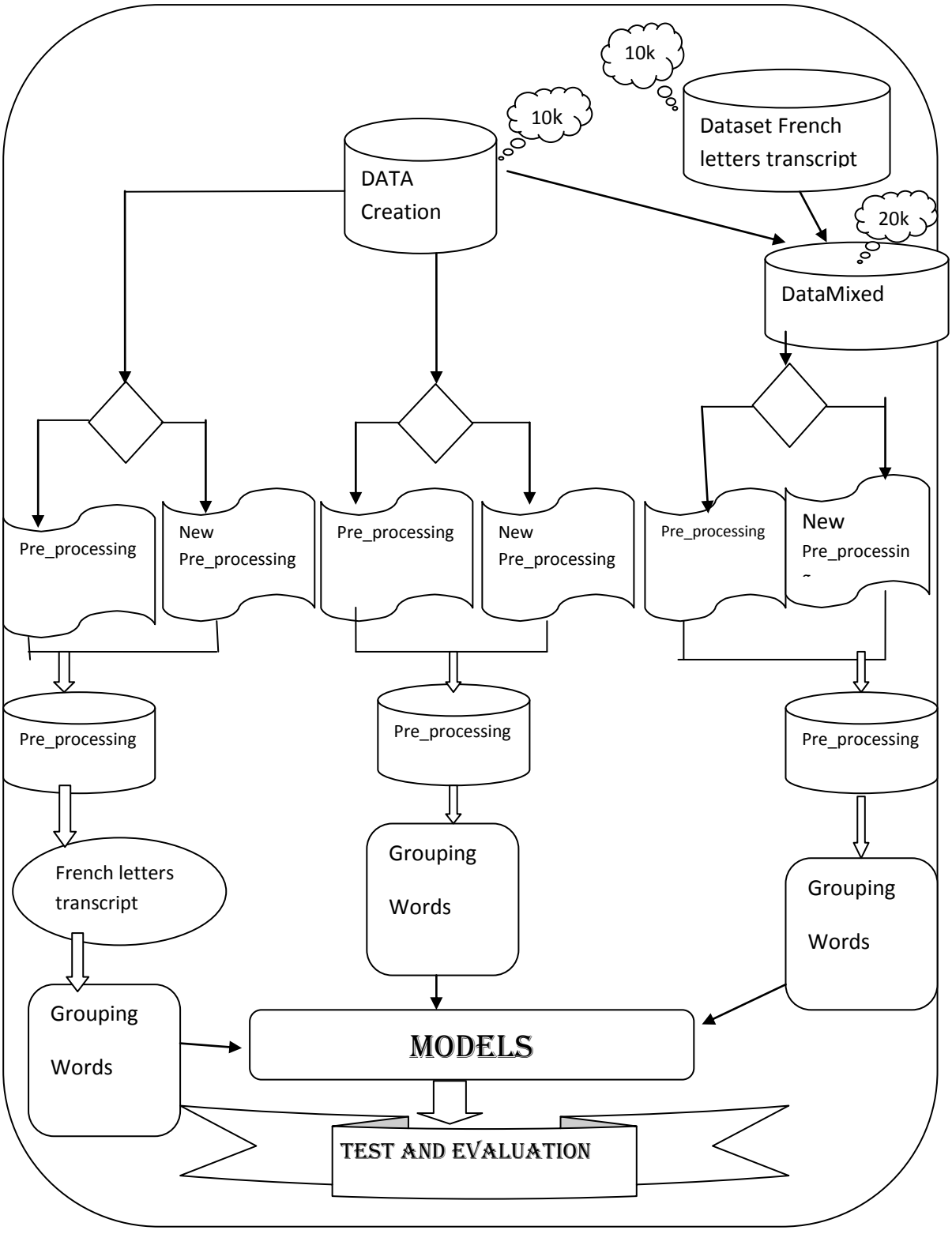


Figure.IV.4. General steps for constructing the annotated dataset 2

THE MODELS

1Models of machine learning (Randomforestclassifier / MultinomialNB / Logistic Regression / SGD Classifier/ SVC classifier)

1. With TF-IDF embedding
2. With Word2Vec embedding

2 Deep learning models

1. Without FastText (LSTM/ GRU/ Bi-LSTM/ Bi-GRU/ CNN)
2. With FastText (LSTM/ GRU/ Bi-LSTM/ Bi-GRU)

3. For transfer learning (Distilbert-base-uncased/ Bert-base-multilingual-uncased/ Bert-base-arabic)

For Bert layer with GRU and LSTM (BERT embedding +LSTM/ BERT embedding +GRU/ BERT embedding +GRU and LSTM)

4. Ensemble learning

ensemble models.

Model1=BERT+LSTM
Model2=BERT+GRU
Model3=BERT+GRU_LSTM
Model4= BERT+LSTM+GRU+GRU_LSTM
Model5=LSTM+GRU_LSTM
Model6=BERT+GRU_LSTM
Model7=LSTM+GRU
Model8=BERT+LSTM+GRU
Model9=LSTM+GRU+GRU_LSTM
Model10= BERT+LSTM+GRU_LSTM
Model11=BERT+GRU+GRU_LSTM

Model1+ Model2
Model1+ Model3
Model1+ Model4
Model2+ Model3
Model2+ Model4
Model3+ Model4
Model1+ Model2+ Model3
Model1+ Model3+ Model4
Model1+ Model2+ Model4
Model2+ Model3+ Model4
Addition from Model1 to 4
Addition from Model1 to 5
Addition from Model1 to 6
Addition from Model1 to 7
Addition from Model1 to 8
Addition from Model1 to 9
Addition from Model1 to 10
Addition from Model1 to 11

Figure.IV.5. General steps for models

4. Test results for models

We run the various models on our three datasets (MultiTopic-Dataset)
Old data 14150.
New data 10000 .
Data total 24150 .

4.1.Models of machine learning :

To test the models and determine the accuracy, we separated the dataset into two parts: test and training (test accounting for 30% of the dataset and training accounting for 70%). We employ TF-IDF and Word2Vec, as well as five classification algorithms: RandomForset (RF), multinomial naive bayes (alpha = 1 and fit-prior = true), LogisticRegression (c=0.5), SGDClassifier (SGD, with loss='log' and penalty='l2'), and LinearSVC (LSVC with C='1.0' and kernel = 'rbf'). We were motivated by the categorization algorithm proposed by for their implementation phase (Joshi et al., 2016). We employ the Gensim toolkit31 for the Word2vec model. We use the FastText library supplied by Facebook on Github6 for FastText. We employ a context of 50 words for both Word2vec and FastText to generate 100-word CBOW and SG representations. The following test results were obtained:

4.1.1. With TF-IDF embedding :

Table.IV.1. show comparison of the test results between multiTopic dataset the old data and with the machine learning models once with TF-IDF and once with Word2vec (both CBOW and skip-gram) respectively. We used optimizers with epoch 10.

Table.IV.1. Results of TF-IDF with ML

DATA 24K																		**DATA 10k**									Old data For colleagues14K		
	A1			**A2**			**B1**			**B2**			**C1**			**C2**			**E2**			**F2**			**G2**			**D**		
	Precision	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**
Randomforestclassifier	0.74	0.60	0.69	0.73	0.63	0.70	0.74	0.66	0.70	0.74	0.67	0.72	0.74	0.65	0.71	0.73	0.68	0.71	0.83	0.62	0.77	0.84	0.63	0.78	0.84	0.65	0.79	**0.68**	0.65	0.65
Multinomial NB	0.72	0.78	0.74	0.73	0.78	0.75	0.72	0.79	0.74	0.72	0.79	0.74	0.73	0.79	0.75	**0.74**	**0.79**	**0.75**	0.85	0.75	0.83	0.85	0.75	0.83	**0.86**	**0.77**	**0.84**	0.64	**0.82**	**0.66**
Logistic Regression	0.71	0.75	0.72	0.71	0.75	0.73	0.72	0.75	0.73	0.71	0.73	0.72	0.72	0.75	0.73	0.72	0.77	0.73	0.80	0.73	0.80	0.83	0.71	0.80	0.81	0.74	0.81	0.64	0.79	0.65
SGD Classifier	0.71	0.75	0.72	0.71	0.77	0.73	0.73	0.74	0.73	0.71	0.73	0.72	0.71	0.75	0.72	0.70	0.77	0.72	0.80	0.77	0.81	0.83	0.76	0.82	0.82	0.76	0.81	0.66	0.76	0.66
SVC classifier	0.73	0.75	0.73	0.73	0.74	0.73	0.73	0.76	0.74	0.73	0.74	0.73	0.74	0.76	0.74	0.74	0.77	0.75	0.81	0.77	0.82	0.83	0.76	0.82	0.82	0.79	0.83	0.65	0.74	0.65

4.1.2. With Word2Vec embedding

Table.IV.2. Results With Word2Vec embedding

DATA 24K																			**DATA 10k**									Old data		
	A1			**A2**			**B1**			**B2**			**C1**			**C2**			**E2**			**F2**			**G2**			**D**		
	Precision	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**	**Accuracy**
Randomforestclassifier	0.62	0.61	0.62	0.62	0.59	0.61	0.61	0.61	0.61	0.60	0.61	0.60	0.62	0.64	**0.63**	0.62	0.63	0.62	**0.66**	0.54	**0.67**	0.65	0.55	**0.67**	0.65	0.58	**0.67**	**0.59**	0.64	**0.57**
MultinomialNB	0.51	0.92	0.52	0.51	**0.93**	0.52	0.52	0.89	0.54	0.52	0.87	0.53	0.52	0.87	0.53	0.51	0.90	0.53	0.47	**0.93**	0.49	0.47	0.89	0.5	0.47	0.75	0.52	0.53	0.94	0.53
Logistic Regression	0.58	0.68	0.60	0.57	0.62	0.58	0.57	0.69	0.59	0.569	0.69	0.58	0.58	0.67	0.59	0.57	0.64	0.58	0.55	0.26	0.57	0.57	0.32	0.58	0.55	0.42	0.59	0.56	0.78	0.56
SGD Classifier	0.53	0.92	0.56	**0.66**	0.08	0.52	0.59	0.17	0.52	0.56	0.71	0.57	0.53	**0.93**	0.553	0.52	**0.93**	0.54	0.56	0.08	0.55	0.50	0.02	0.55	0.51	0.84	0.56	0.54	**0.98**	0.54
SVC classifier	0.58	0.67	0.60	0.58	0.62	0.58	0.57	0.67	0.59	0.57	0.67	0.58	0.58	0.67	0.59	0.58	0.64	0.58	0.55	0.31	0.57	0.58	0.37	0.59	0.56	0.43	0.59	0.57	0.74	0.56

. With TF-IDF embedding Accuracy

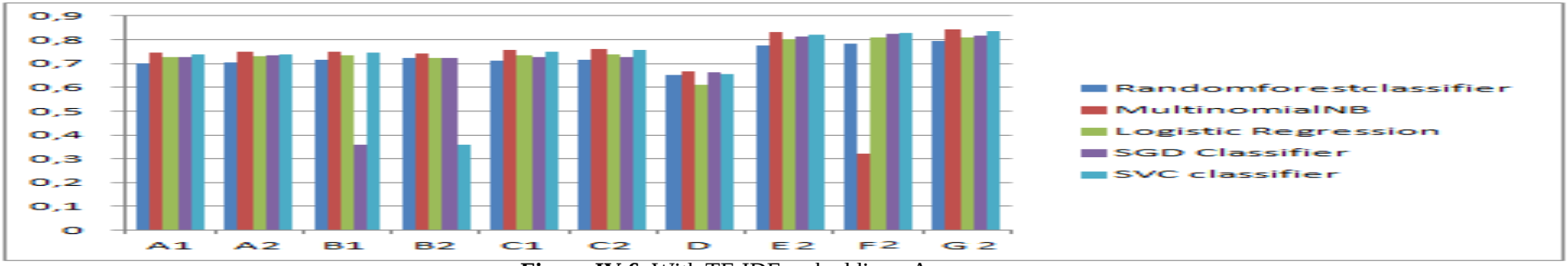


Figure.IV.6. With TF-IDF embedding Accuracy

.With Word2Vec embedding Accuracy

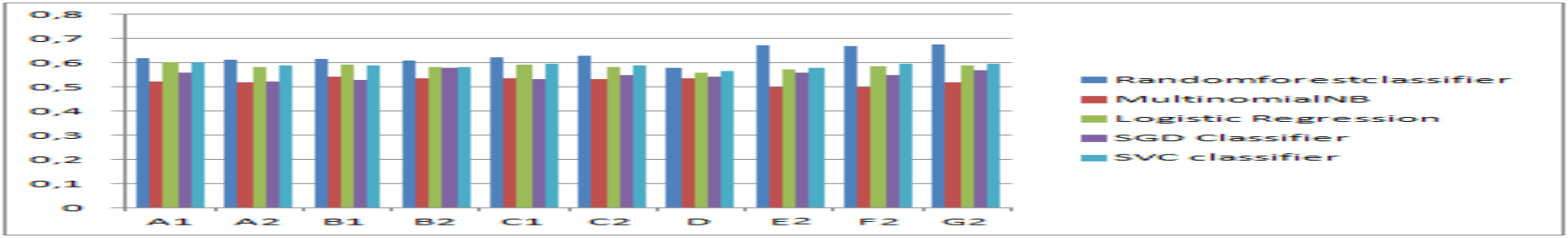


Figure.IV.7. With Word2Vec embedding Accuracy

Discussion:

- We've noticed that TF-IDF embedding works better with machine learning in each classifier than Word2Vec embedding.
- the best accuracy with TF-IDF is 0.844 for 10 k Arabic new Pre-processing , 0.759 with 24k and 0.665 with old data
- the best accuracy with W2V is 0.676 for 20 k Arabic new Pre-processing, 0.632 with 24k and 0.578 with old data
- It's also worth noting that the 10k dataset outperforms the 24k dataset and old data.

4.2.Deep Learning Models :

We also divided the dataset into 30 percent for testing and 70 percent for training for deep learning models. Table.IV.3/ Table.IV.4 shows the test results for the multi-Topic dataset and Old data after comparing the performance of several deep learning models with and without FastText. The first layer in LSTM, GRU, Bi-LSTM, and Bi-GRU is a randomly-initialized word embedding layer that converts words in sentences into feature maps. FastText is used to build the embedding matrix weights, and then the (LSTM/GRU/ Bi-LSTM / Bi-GRU) layer scans the feature map. A Dropout layer with a probability of 0.5 is added to reduce over-fitting. The scores are then sent into a single feed-forward (fully-connected) layer with Relu activation, and the output of that layer is fed into a sigmoid layer that predicts the output classes. With epoch 10, we employed Adam optimizers. The first layer in CNN is a randomly initialized word embedding layer, which converts words in sentences into feature maps. FastText is used to construct the embedding matrix weights. The feature map is then scanned by the CNN layer. To get the highest score for each pattern, global maxpooling is done to the output layer. The scores are then sent into a single feed-forward (fully-connected) layer with Relu activation, and the output of that layer is fed into a sigmoid layer that predicts the output classes. With epoch 10, we employed Adam optimizers.

4.2.1.Without FastText

Table.IV.3. Test results of datasets without FastTaxt

DATA 24K												**DATA 10k**						**Old data For colleagues 14K**	
	A1		**A2**		**B1**		**B2**		**C1**		**C2**	**E2**		**F2**		**G2**		**D**	
	f1-score	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**
LSTM	0.74	0.73	0.73	0.72	0.73	0.73	0.72	0.72	0.72	0.74	0.74	0.74	0.80	0.83	0.80	0.81	**0.87**	**0.85**	0.67	0.64
GRU	0.74	0.73	0.73	0.72	0.73	0.73	0.73	0.73	0.74	0.74	0.74	0.74	0.81	0.83	0.76	0.8	0.83	0.85	0.66	0.65
Bi-LSTM	0.73	0.73	0.69	0.71	0.70	0.72	0.71	0.71	0.74	0.74	0.73	0.73	0.79	0.83	0.78	0.81	0.79	0.83	0.69	0.65
Bi-GRU	0.68	0.72	0.72	0.71	0.72	0.73	0.70	0.71	**0.75**	**0.75**	0.73	0.74	0.79	0.81	0.78	0.8	0.8	0.83	**0.70**	**0.66**
CNN	0.33	0.51	0.33	0.50	0.33	0.51	0.33	0.49	0.33	0.50	0.33	0.51	0.36	0.56	0.35	0.53	0.35	0.55	0.35	0.55

4.2.2.With FastText

Table.IV.4. Test results of datasets with FastTaxt

DATA 24K												**DATA 10k**						Old data For colleagues 14K	
	A1		**A2**		**B1**		**B2**		**C1**		**C2**		**E2**		**F2**		**G2**		**D**	
	f1-score	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**	**f1-score**	**accuracy**
LSTM	0.70	0.72	0.72	0.72	0.71	0.71	0.71	0.71	0.72	0.72	0.70	0.71	0.81	0.83	0.78	0.80	0.80	0.82	0.64	0.64
GRU	0.71	0.72	0.72	0.71	0.72	0.71	0.71	0.71	0.74	0.71	0.72	0.71	0.81	0.83	0.78	0.80	0.81	0.83	**0.68**	**0.66**
Bi-LSTM	0.71	0.71	0.73	0.72	0.73	0.71	**0.74**	0.72	0.71	0.72	0.73	**0.73**	**0.82**	**0.84**	0.8	0.81	0.8	0.82	0.67	0.66
Bi-GRU	0.73	0.72	0.72	0.72	0.72	0.71	0.72	0.71	0.73	0.72	0.72	0.72	**0.82**	**0.84**	0.79	0.80	0.81	0.83	0.66	0.66

Without FastText Accuracy

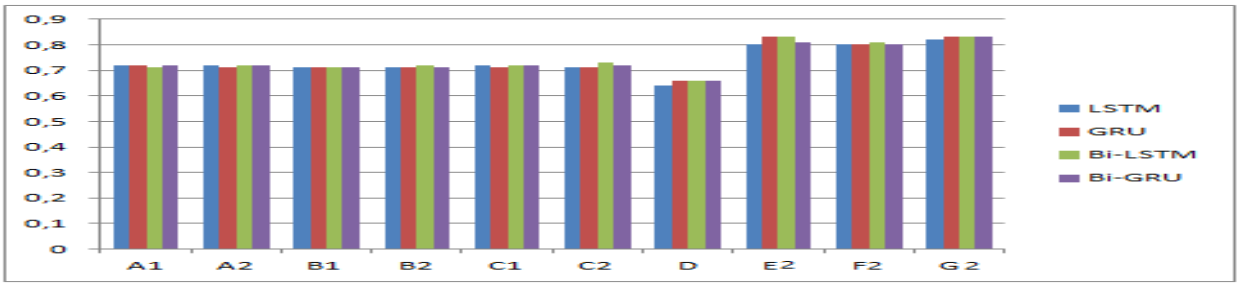


Figure.IV.8. Without FastText Accuracy

With FastText Accuracy

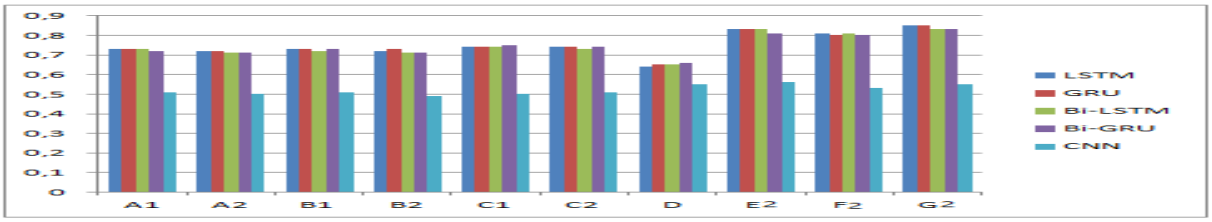


Figure.IV.9. with FastText Accuracy

Discussion :

We can observe deep learning models . We can point out that CNN is not as good as other models.

We can also see that LSTM and GRU perform better than Bilstm and Bi-GRU in all dataset. the best accuracy was 66% for bi-lstm and bi-gru with fasttext for old dataset , for 10k dataset LSTM with FastText gives the best result with an accuracy of 84% and 74% for 24k dataset. We can notice that Bi-Lstm and Bi-GRU perform better with fastText.

We notice that FastText embeddings has improved the accuracy and f1_score in both datasets.

4.3. For transfer learning :

We made the decision to fine-tune Hugging Face transformers' Bert-base-arabic, Bert-base-multilingual-uncased, and distilbert-base-uncased pre-trained models. we got the following test result in Table.IV.5. We used optimizers with epoch 5.

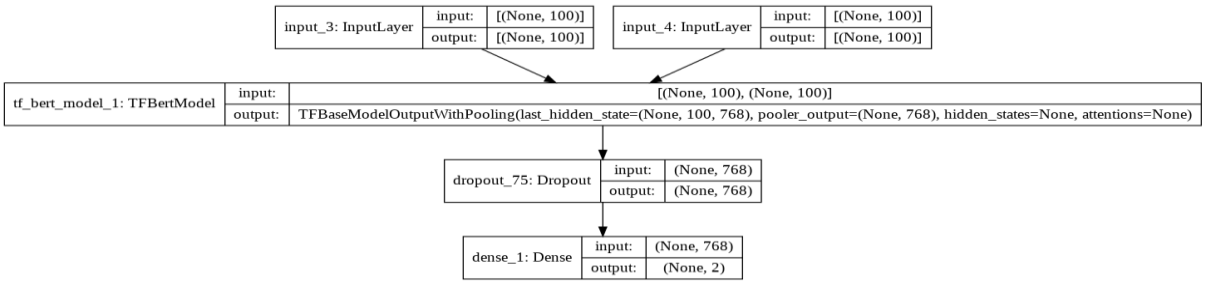


Figure.IV.10. BERT architecture

Table.IV.5. Fine-tune results

DATA 24K												**DATA 10k**												Old data For colleagues 14K	
	A1		**A2**		**B1**		**B2**		**C1**		**C2**		**E1**		**E2**		**F1**		**F2**		**G1**		**G2**		**D**	
	f1-score	accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	Accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy
Distilbert-base-uncased	0.69	0.68	0.68	0.68	0.68	0.67	0.67	0.63	0.69	0.68	0.69	0.69			0.73	0.74			0.75	0.75			0.76	0.76	0.67	0.67
Bert-base-multilingual-uncased	0.76	0.75	0.75	0.75	0.72	0.72	0.71	0.70	0.74	0.74	0.74	0.74			0.79	0.79			0.78	0.78			0.83	0.83	0.68	0.68
Bert-base-arabic	0.74	0.76	**0.76**	**0.76**	**0.76**	**0.76**	**0.76**	**0.76**	0.75	0.75	0.73	0.74	0.82	0.84	0.83	0.84	**0.85**	**0.85**	0.83	0.84	0.82	0.84	0.81	0.82	**0.72**	**0.71**

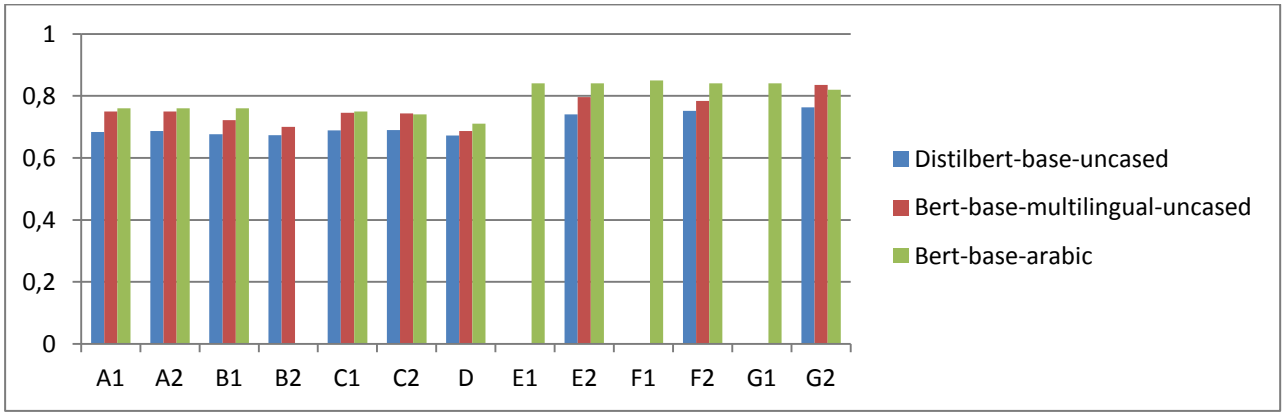


Figure.IV.11. Distilbert-base-uncased/ Bert-base-multilingual-uncased/ Bert-base-arabic Accuracy

Discussion:

- We see that the performance of pretrained language models on new dataset varies accuracy from 73% to 85% and old dataset varies accuracy from 67% to 72% and total dataset varies accuracy from 67% to 76%.
- The best accuracy was 85% for Bert base Arabic for new dataset, and f1-score of 85%.
- We notice that the performance of pretrained language models achieved highly better results on both datasets.

4.3.1. Bert layer with GRU and LSTM :

After further tuning, we can utilize the BERT model as a layer in the Tensorflow.keras library to build a model. Bert with LSTM and Bert with GRU architectures will be presented in the following section.

In each time step, we utilised all of the hidden states produced by the recurrent layers (GRUs or LSTMs). In an RNN layer, employing all hidden states results in a concatenation of two vectors, the first for the forward RNN layer and the second for the backward RNN layer, with the size of the number of units employed in our instance 192. This is demonstrated in Figure.IV.12, 13, and 14, where the outputs of the first and second RNN layers, each of which has a size of (100, 384), are concatenated and then sent through a Dropout layer before being transmitted to the final dense layer. The results are shown in Table.IV.6. Figure.IV.12, 13, and 14 show the architecture of each model. Figure.IV.14shows the comparison of the two sets of data.

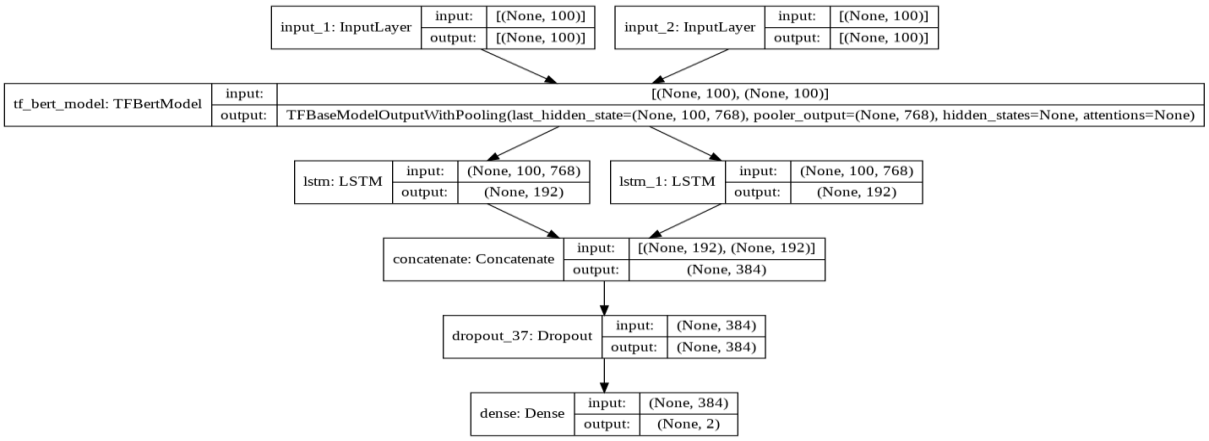


Figure.IV.12. The Bert with LSTM architecture

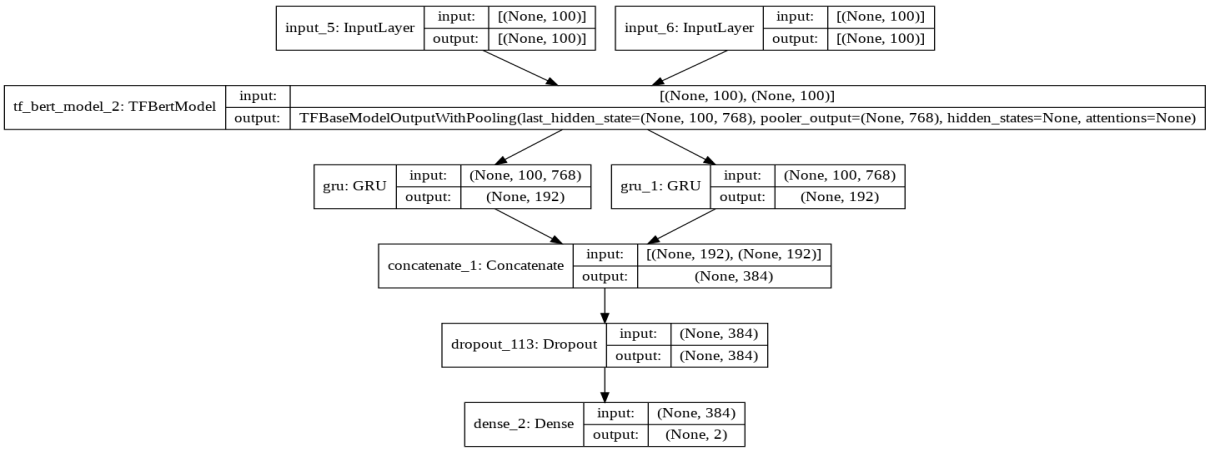


Figure.IV.13. Bert with GRU architecture

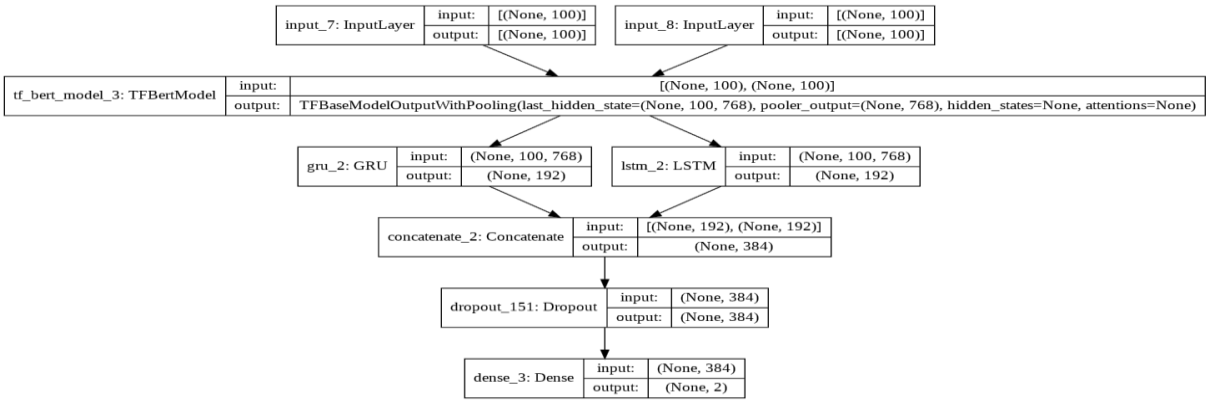


Figure.IV.14. Bert with GRU and LSTM

Table.IV.6. Bert embedding with GRU and LSTM results

DATA 24K												DATA 10k												Old data For colleagues 14K	
	A1		A2		B1		B2		C1		C2		E1		E2		F1		F2		G1		G2		D	
	f1-score	accuracy	f1-score	Accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy
BERT embedding +LSTM	0.78	0.77	0.75	0.76	0.76	0.76	0.75	0.75	0.75	0.75	0.73	0.73	0.82	0.84	0.83	0.83	0.84	0.86	0.83	0.84	0.79	0.82	0.82	0.83	**0.74**	0.70
BERT embedding +GRU	0.75	0.76	0.76	0.76	**0.78**	**0.77**	0.72	0.72	0.75	0.73	0.72	0.73	**0.86**	**0.87**	0.83	0.84	0.83	0.85	0.84	0.84	0.80	0.83	0.80	0.81	0.73	0.70
BERT embedding +GRU and LSTM	0.75	0.76	0.75	0.75	0.74	0.75	0.75	0.75	0.74	0.74	0.73	0.73	0.84	0.86	0.84	0.85	0.83	0.85	0.84	0.84	0.80	0.83	0.82	0.82	0.70	**0.71**

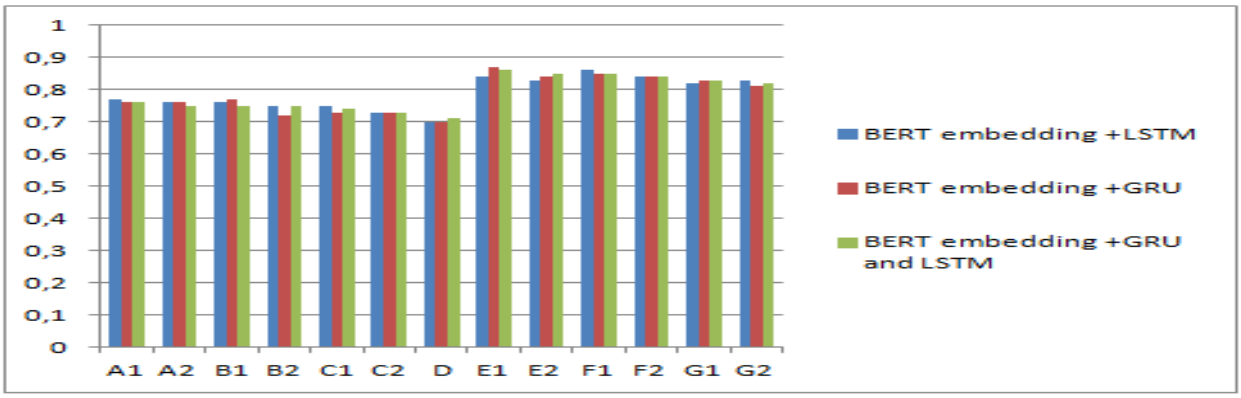


Figure.IV.15. Accuracy Bert embedding with GRU and LSTM results

Discussion:

- We got the following results after running our models on the three datasets:
- Firstly, the record of f1-score is confined between 70.% and 87%, and accuracy is confined between 72% and 87% in all data .also the high score was hit by the BERT embedding +GRU model with 87% for new data, and 77% with total dataset .
- BERT embedding +LSTM and Bert embedding +GRU and LSTM They're the best models in the old data, is confined between 70% and 74% .
- Then, the BERT model with embedding +GRU was the best in accuracy with 87%

4.4. Ensemble learning :

In this section, we suggest constructing ensemble learning models using the BERT model as well as other deep learning architectures based on LSTMs and GRUs. BERT with LSTM, Bert with GRU, and Bert with both LSTM and GRU are examples of combinations.

To create our ensemble learning models, we individually train the models that make up the ensemble model on the training set. Then, in order to classify a new instance, we integrate the outputs of each ensemble model by taking the mean for the ensemble learning model output. We simply add the outputs of the ensemble models and divide the result by the number of these models to get the ordinary mean of outputs.

Model1=BERT+LSTM
Model2=BERT+GRU
Model3=BERT+GRU_LSTM
Model4= BERT+LSTM+GRU+GRU_LSTM
Model5=LSTM+GRU_LSTM
Model6=BERT+GRU_LSTM
Model7=LSTM+GRU
Model8=BERT+LSTM+GRU
Model9=LSTM+GRU+GRU_LSTM
Model10= BERT+LSTM+GRU_LSTM
Model11=BERT+GRU+GRU_LSTM

DATA 24K												DATA 10k												Old data For colleagues 14K	
	A1		A2		B1		B2		C1		C2		E1		E2		F1		F2		G1		G2		D	
	f1-score	accuracy	f1-score	accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy
Model1	0.77	0.77	0.76	0.77	0.77	0.77	0.76	0.77	0.76	0.76	0.74	0.75	0.84	0.85	0.84	0.85	0.85	0.86	0.83	0.84	0.82	0.84	0.83	0.84	0.74	0.72
Model2	0.76	0.76	0.76	0.77	0.78	0.77	0.76	0.77	0.76	0.75	0.74	0.75	0.84	0.86	0.85	0.86	0.85	0.86	0.84	0.85	0.82	0.85	0.82	0.83	0.73	0.71
Model3	0.77	0.77	0.76	0.77	0.77	0.77	0.77	0.77	0.75	0.75	0.74	0.75	0.84	0.86	0.84	0.85	0.85	0.86	0.84	0.85	0.82	0.84	0.83	0.84	0.71	0.71
Model4	0.77	0.77	0.76	0.77	0.78	0.78	0.77	0.78	0.76	0.76	0.74	0.75	0.85	0.86	0.84	0.85	0.86	0.87	0.85	0.86	0.82	0.84	0.84	0.84	0.74	0.72
Model5	0.76	0.76	0.76	0.76	0.76	0.77	0.77	0.77	0.75	0.75	0.74	0.74	0.84	0.86	0.83	0.84	0.85	0.87	0.84	0.85	0.80	0.83	0.83	0.84		
Model6	0.77	0.77	0.76	0.77	0.77	0.77	0.77	0.77	0.75	0.75	0.74	0.75	0.84	0.86	0.84	0.85	0.85	0.86	0.84	0.85	0.82	0.84	0.83	0.84		
Model7	0.77	0.77	0.76	0.77	0.78	0.78	0.75	0.75	0.74	0.75	0.74	0.74	0.8	0.85	0.84	0.85	0.85	0.86	0.84	0.84	0.79	0.83	0.82	0.83		
Model8	0.77	0.78	0.76	0.77	0.77	0.78	0.76	0.77	0.75	0.76	0.74	0.75	0.85	0.87	0.84	0.84	0.85	0.86	0.84	0.85	0.82	0.84	0.83	0.84		
Model9	0.77	0.78	0.76	0.77	**0.78**	**0.78**	0.76	0.77	0.75	0.75	0.74	0.74	0.85	0.86	0.85	0.85	0.85	0.86	0.85	0.85	0.80	0.83	0.83	0.84		
Model10	0.77	0.77	0.76	0.77	0.77	0.78	0.77	0.78	0.76	0.76	0.75	0.75	0.85	0.86	0.84	0.85	0.85	0.87	0.85	0.86	0.82	0.84	0.83	0.84		
Model11	0.76	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.76	0.76	0.74	0.74	**0.86**	**0.87**	0.85	0.85	0.85	0.86	0.85	0.85	0.82	0.84	0.83	0.84		

Table.IV.7.resultas of ensemble models 1

DATA 24K												DATA 10k												Old data For colleagues 14K	
	A1		A2		B1		B2		C1		C2		E1		E2		F1		F2		G1		G2		D	
	f1-score	accuracy	f1-score	accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	Accuracy	f1-score	accuracy	f1-score	accuracy
Model1+ Model2	0.77	0.76	0.76	0.76	0.77	0.77	0.76	0.76	0.75	0.75	0.74	0.74	0.85	0.86	0.85	0.85	0.86	0.86	0.84	0.85	0.84	0.84	0.83	0.83	0.71	0.72
Model1+ Model3	0.77	0.77	0.76	0.76	0.76	0.76	0.77	0.77	0.75	0.75	0.75	0.75	0.85	0.85	0.85	0.85	0.86	0.86	0.84	0.84	0.84	0.84	0.83	0.83	0.71	0.72
Model1+ Model4	**0.78**	**0.78**	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.84	0.85	0.86	0.86	0.84	0.85	0.84	0.84	0.83	0.83		
Model2+ Model3	0.76	0.76	0.76	0.76	0.76	0.76	0.77	0.77	0.75	0.75	0.7	0.74	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.83	0.84	0.83	0.83	0.71	0.71
Model2+ Model4	0.76	0.76	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.74	0.74	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.84	0.83	0.83		
Model3+ Model4	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.84	0.83	0.84		
Model1+ Model2+ Model3	0.77	0.77	0.76	0.76	0.76	0.76	0.77	0.77	0.75	0.75	0.74	0.74	0.85	0.85	0.85	0.85	0.86	0.86	0.85	0.85	0.83	0.83	0.83	0.83		
Model1+ Model3+ Model4	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.85	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.84	0.83	0.84		
Model1+ Model2+ Model4	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.84	0.83	0.83		

Model2+ Model3+ Model4	0.76	0.76	0.77	0.77	0.76	0.76	0.77	0.77	0.75	0.75	0.74	0.74	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.84	0.80	0.83	0.83		
Addition from Model1 to 4	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.85	0.83	0.84	0.83	0.83	**0.72**	**0.72**
Addition from Model1 to 5	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	**0.86**	**0.87**	0.86	0.86	0.84	0.80	0.83	0.84		
Addition from Model1 to 6	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	0.86	0.86	0.86	0.86	0.84	0.84	0.84	0.84		
Addition from Model1 to 7	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	**0.86**	**0.87**	0.85	0.86	0.83	0.84	0.83	0.84		
Addition from Model1 to 8	0.77	0.77	0.76	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.85	0.85	0.86	0.86	0.85	0.86	0.84	0.84	0.83	0.83		

Addition from Model1 to 9	0.77	0.77	0.76	0.76	**0.78**	**0.78**	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.84	0.85	0.86	0.86	0.85	0.85	0.84	0.84	0.83	0.84		
Addition from Model1 to 10	0.77	0.77	0.77	0.76	0.77	0.77	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.84	0.85	0.86	0.86	0.85	0.86	0.83	0.84	0.83	0.84		
Addition from Model1 to 11	0.77	0.77	0.76	0.76	0.78	0.78	0.77	0.77	0.75	0.75	0.75	0.75	0.86	0.86	0.84	0.85	**0.86**	**0.87**	0.85	0.85	0.84	0.84	0.83	0.84		

Table.IV.8. results of ensemble models 2

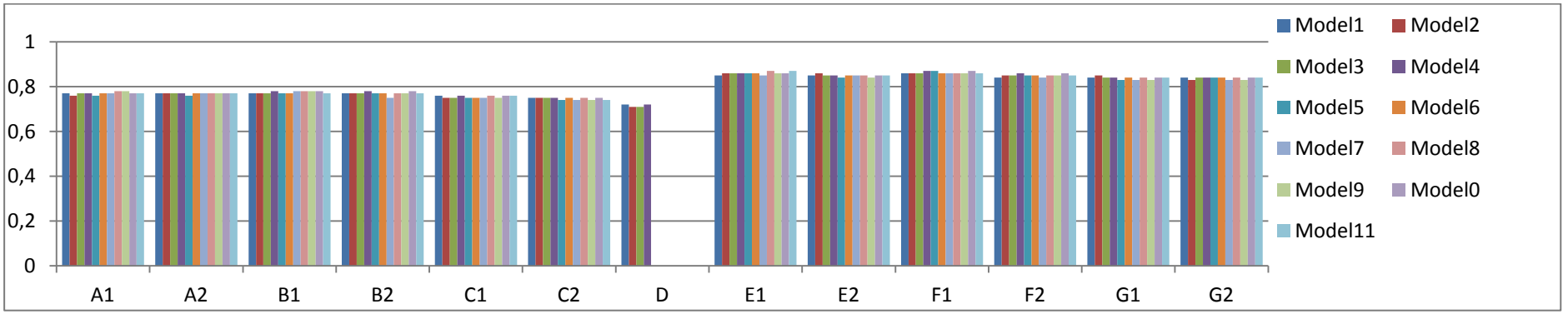


Figure.IV.16. Ensemble learning Accuracy **1**

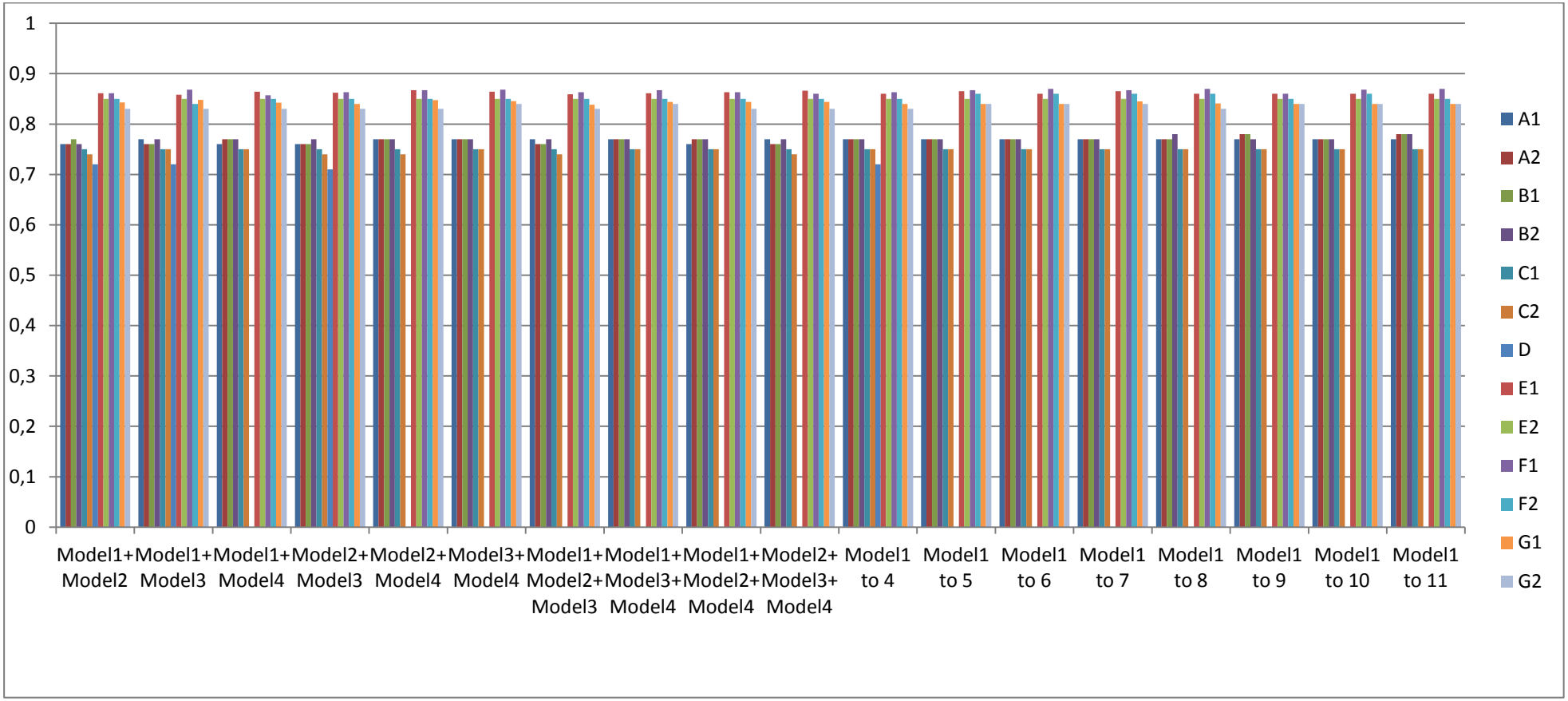


Figure.IV.17. Ensemble learning Accuracy **2**

Discussion :

- For ensemble learning we can see that :
- Model 4and 5 and 11 with an accuracy of 87% and f1_score of 86%. in 10k arabicfr (Pre-processing)
- Model 10 with an accuracy of 87% and f1_score of 85% in 10k arabic (Pre-processing)
- In data collection(24k/48k) with (Pre-processing /new Pre-processing) accuracy betwin 75%and 78% and f1_score of 74%and78%
- Ensembles models have improved the best results with 78% accuracy and 78% f1-score in data 24k ,48k with (Pre-processing /new Pre-processing) especially in (Addition from Model1 to 9/Addition from Model1 to 11/Model1 + Model4)
- The best results with 86% accuracy and 87% f1-score in data 10k ,20k with (Pre-processing /new Pre-processing) especially in (Addition from Model1 to 11)

4.5. Comparison between (deep learning models and pretrained language models) And General (f1_score/ accuracy) comparison :

In the table we selected the best resultas of f1_score and accuracy for all models and selected data 10k because it gave us the best results and compared them with old data.

Table.IIV.9. Comparison between (deep learning models and pretrained language models) And General (f1_score/ accuracy) comparison

DATA 10K												
	Data10k Arabic				**Data10K ArabicF**		**Data20k arabicFR**		Old data For colleagues 14K		**Percentage**	
	E1		**E2**		**F1**		**G2**					
	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy	f1-score	accuracy
LSTM							**0.87**	**0.85**	0.67	0.64	**20 %**	**21%**
GRU							0.83	0.85	0.66	0.65	17%	20%
Bi-LSTM							0.79	0.83	0.69	0.65	10%	18%
Bi-GRU							0.80	0.83	0.69	0.65	11%	18%
CNN			0.36	0.56					0.35	0.55	1%	1%
Distilbert-base-uncased							0.76	0.76	0.67	0.67	9%	9%
Bert-base-multilingual-uncased							0.83	0.83	0.68	0.68	15%	15%
Bert-base-arabic					0.85	0.85			0.72	0.71	13%	14%
BERT embedding +LSTM					0.84	0.86			0.74	0.70	10%	16%
BERT embedding +GRU	0.86	0.87							0.73	0.70	13%	17%
BERT embedding +GRU and LSTM	0.84	0.86							0.70	0.71	14%	15%

4.6.General f1_score comparison between deep learning models and pretrained language models :

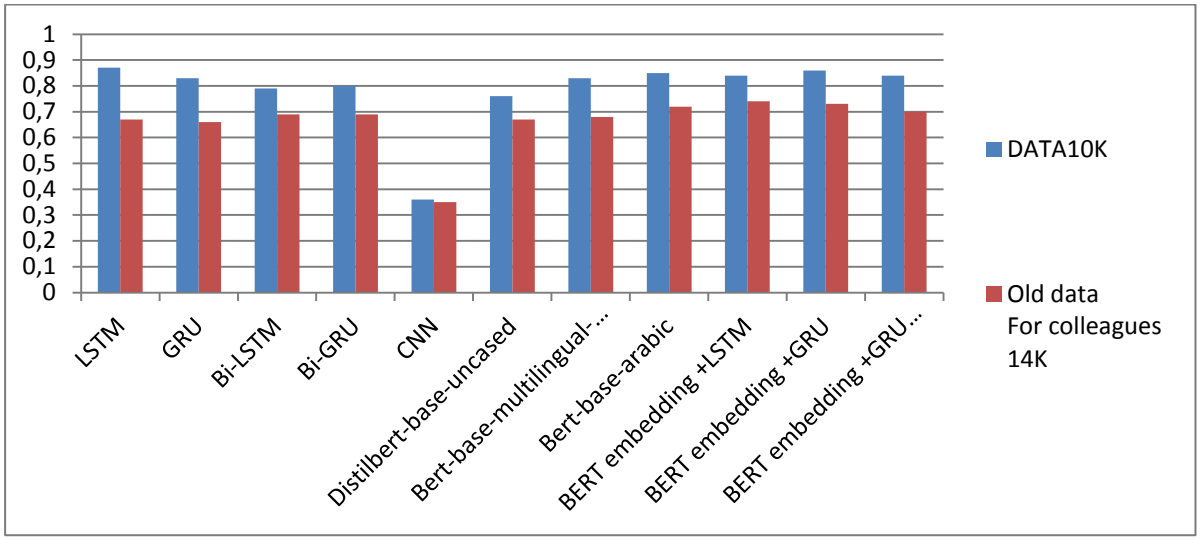


Figure.IV.18. General f1_score comparison between deep learning models and pretrained language models

4.7.General accuracy comparison between deep learning models and pretrained language models :

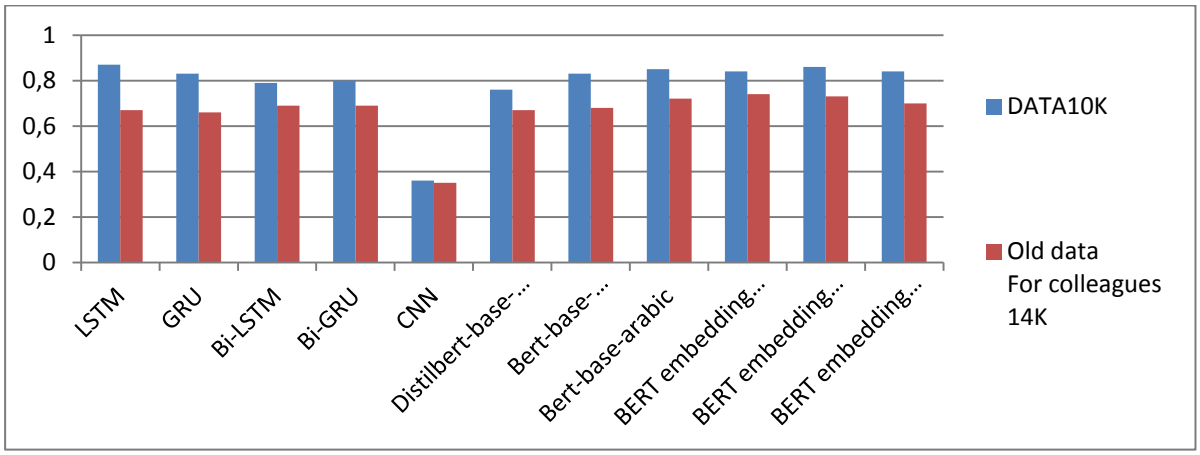


Figure.IV.19. General Accuracy comparison between deep learning models and pretrained language models

General discussion

- We notice in the first ML models that TF-IDF perform better than Word2vec, the best TF-IDF results was with SGD classifier 84% accuracy.
- For deep learning models we can see that the RNN models perform better than CNN models, the best RNN model is LSTM which gives 85% accuracy and 87% F1-Score.
- We can also see that LSTM and GRU perform better then Bi-LSTM and Bi-GRU .
- The best accuracy was 84% and F1-Score 82% for bi-lstm and bi-gru with FastText
- The best accuracy was 85% and F1-Score 87% for LSTM Without FastText
- We can notice that Bi-Lstm and Bi-GRU perform better with fastText.
- The Bert-base-Arabic model achieved the best results with an accuracy of 85% and f1_score of 85% in our dataset.
- Ensembles models have improved the best results with 78% accuracy and 78% f1-score in data 24k ,48k with (Pre-processing /new Pre-processing) especially in (Addition from Model1 to 9/Addition from Model1 to 11/Model1 + Model4)

- We note that data creation achieved very better results in both data sets, after data collection and fineli old data.
- Another conclusion is that merging various high-performing classifiers that have been trained individually improves the prediction capacity of the resulting ensemble learning classifier.
- Data creation gives the best result with an accuracy of 87% and f1_score of 86% in (BERT embedding +GRU) after with an accuracy of 85% and f1_score of 87% in

  (LSTM).
- We notice that ensemble models have improved the accuracy and f1_score in both datasets.
- The best models is transfer learning (Bert layer with GRU) afetr Deep learning models RNN(LSTM).

General Conclusion

We wanted to improve our capacity to recognize hate speech in multilanguages, particularly in Algerian dialects, because there have been few studies in this language. Designing gadgets that detect hate speech is a huge accomplishment in and of itself, since it results in a considerable vaccination of the level of speech and shields the reader from hate speech language.

In Our project makes a concerted effort to combat the propagation of hate speech in Algerian dialect on social media. We proposed a strategy for coping with the language's issues.

First, we compiled a hate speech dataset by combining 10000 comments from social media sites like Facebook, YouTube, and Twitter with historical data. The total number of records is 24,150. (12075 positive and 12077 negative comments). . These comments are from different topics such as politic, misogyny, racism and many others.

Second, we use two different methods to process the information:

In addition to employing transcript and grouping, the first and second methods are pretreatment and new pretreatment. We proposed to filter the comments, tokenization, removing duplicate letters, and removing punctuation and special characters…

We now have three data sets as a result of the processing.

The first set of data (Data creation) is new information that we have gathered.

The old data is the second set of information.

The total data (Data collection) is the third data, which combines the first and second.

Then, in order to improve our dataset, we run it through a series of machine learning, deep learning, and transfer learning algorithms.

We used tf-idf and w2v in machine learning, where we found that tf-idf is better than w2v. The best result we reached with machine learning was with the tf-idf SGD classifier with 84% accuracy for the new dataset and 71% accuracy with the old dataset. The best models is transfer learning (Bert layer with GRU) afetr Deep learning models RNN(LSTM).

We also tested it on deep learning models, and the best result was 74% for the new data with Bi-LSTM and Bi-GRU with FastText embedding.

We also intend to test our models on large data in the future, as the data we utilized was insufficient for learning. In the collecting phase, we'll try to avoid comments with the same meaning, and we'll try to utilize other pre-trained models with more robust architecture, and last but not least, we'll try to develop a pre-trained model for the Algeria dialect. In addition to generalizing it to all Algerian dialects (since dialects vary by location, such as Tamazighit, Chaoui, Chalhi, Taraki, Mezabe, etc.), and to all languages in the globe.

Bibliographie

A. Alakrot, L. M. (2018). Nikolov, “Dataset Construction for the Detection of Anti-Social Behaviour in Online Communication in Arabic,” Procedia Comput. Sci., vol. 142,. pp. 174–181.

A. Alakrot, L. Murray, and N. S. Nikolov,. (2018). “Towards Accurate Detection of Offensive Language in Online Communication in Arabic,” Procedia Comput. Sci., vol. 142, 2018. . pp. 315–320, .

A. H. Alduailej and M. B. Khan. (2017). “The challenge of cyberbullying and its automatic detection in Arabic text,” in 2017 International Conference on Computer and Applications (ICCA), 2017,. pp. 389–394. .

Abad et all . (2016). Advances in Speech and Language Technologies for Iberian Languages: Third International Conference, IberSPEECH 2016, Lisbon, Portugal, November 23-25, 2016, Proceedings. Cham:.

Abozinadah et al, A. V. (2015). “Detection of abusive accounts with Arabic tweets,” Int. J. Knowl. Eng., vol. 1, no. 2,. pp. 113–119,.

Affairs, C. (2020). .Requested by the FEMM committee Violence against Women Psychological violence and. April.

Albadi et al, M. K. (2018). “Are they Our Brothers? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere,” 2018 IEEE/ACM Int. Conf. Adv. Soc. Networks Anal. Min., 2018. pp. 69–76,.

Alfina et al, R. M. (2018). , “Hate speech detection in the Indonesian language: A dataset and preliminary study,” 2017 Int. Conf. Adv. Comput. Sci. Inf. Syst. ICACSIS 2017, vol.2018–Janua, no. October, pp. 233–237, 2018. .

Alhuzali and M. Abdul-Mageed et al, A. (2018). “Think Before Your Click: Data and Models for Adult Content in Arabic Twitter,” in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), .

Alkiviadou, N., Mchangama, J., & Tool, R. (1994). *GLOBAL HANDBOOK ON HATE SPEECH LAWS REBUILDING THE BULWARK OF LIBERTY GLOBAL HANDBOOK ON HATE SPEECH LAWS.*

Allhorn, A., Fachbereich, d., & Dr-Ing habil Alexander Löser Beuth, G. *Transfer Learning for Hate Speech Detection.*

Alpaydin, E. (2020). Introduction to Machine Learning. MIT Press. https://books.google.dz/books?id=tZnSDwAAQBAJ.

Andrew,Ng. (2013). “Machine Learning and AI via Brain simulations”. Stanford University. https://ai.stanford.edu/~ang/slides/DeepLearning-Mar2013.pptx.

B. Gambäck and U. K. Sikdar. (2017). “Using Convolutional Neural Networks to Classify Hate-Speech,” Assoc. Comput. Linguist., no. 7491, pp. 85–90, 2017. .

B. Haidar, M. C. (2017). “A Multilingual System for Cyberbullying Detection : Arabic Content Detection using Machine Learning,” Adv. Sci. Technol. Eng. Syst. J., vol. 2, no. 6, .

B.A. Toms, et al. (2020). Physically interpretable neural networks for the geosciences: Applications to earth system variability.

Bahdanau, D., Cho, K., & Bengio, Y. (2015). *NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE*.

Burnap & Williams. (2014). “Hate Speech, Machine Classification and Statistical Modelling of Information Flows on Twitter: Interpretation and Communication for Policy Decision Making,” in Proceedings of the Conference on the Internet, Policy & Politics, 2014, pp. 1–18.

Burnap, P., & Williams, M. . (2016). Us and Them: Identifying Cyber Hate on Twitter across Multiple Protected Characteristics. EPJ Data Science, 5(1), 1–15. https://doi.org/10.1140/epjds/s13688-016-0072-6.

Campbell, J. C. Health consequences of intimate partner violence. (2002). Health consequences of intimate partner violence. The lancet, 359(9314), 1331-1336. 359(9314), 1331-1336.

Chakrabarty, T. G. (2019). . Pay “Attention” to your Context when Classifying Abusive Language. 2017, 70–79. https://doi.org/10.18653/v1/w19-3508.

Cho et al . (2014). Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014.

Chorowski et al . (2015). Attention-based models for speech recognition. Advances in neural information processing systems, 28.

Creswell, A. A. (2017). On denoising autoencoders trained to minimise binary cross-entropy. arXiv preprint arXiv:1708.08487.

DAHAH, R. L. (2020). La Détection de la colère chez le conducteur en utilisant le Deep Learning.

Davidson, T. W. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. . *In ICWSM* , (pp. 1–4).

Del Vigna, A. C. (2017). “Hate me, hate me not: Hate speech detection on Facebook,” CEUR Workshop Proc., vol. 1816, pp. 86–95, 2017.

Dessí et al . (2020). Dessí, D., Dragoni, M., Fenu, G., Marras, M., & Reforgiato Recupero, D. Deep Learning Adaptation with Word Embeddings for Sentiment Analysis on Online Course Reviews. In Deep Learning-Based Approaches for Sentiment Analysis (pp. 57–83). Springer.

Devlin and Chang. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.

Dinakar et al, R. R. (2011). “Modeling the detection of Textual Cyberbullying.,” Soc. Mob. Web, vol. 11, no. 02, 2011. . pp. 11–17,.

E. A. Abozinadah and J. H. Jones. (2017). “A Statistical Learning Approach to Detect Abusive Twitter Accounts,” Proc. Int. Conf. Comput. Data Anal. - ICCDA ’17, pp. 6–13, 2017. .

E. Abdelfatah et al, G. T. (2017). “UNSUPERVISED DETECTION OF VIOLENT CONTENT IN ARABIC SOCIAL MEDIA,” Comput. Sci. Inf. Technol. (CS IT), pp. 1–7, .

Fernandez & Alani. (2018). “Contextual semantics for radicalisation detection on Twitter,” CEUR Workshop Proc., vol. 2182, 2018. .

Fischer, Thomas, and Christopher Krauss. (2018). “Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions.” European Journal of Operational Research, vol. 270, no. 2, 2018, doi:10.1016/j.ejor.2017.11.054. 654-669.

G. K. Pitsilis, H. R. (2018). “Effective hate-speech detection in Twitter data using recurrent neural networks,” Appl. Intell., vol. 48, no. 12, pp. 4730–4742, Dec. 2018. .

G. Xiang et al, B. F. (2012). “Detecting offensive tweets via topical feature discovery over a large scale twitter corpus,” Proc. 21st ACM Int. Conf. Inf. Knowl. Manag. - CIKM ’12, . p. 1980,.

Géron, A. (2017). Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. O’Reilly Media, Inc.

Gers, F. A. (20020.). Learning precise timing with LSTM recurrent networks. Journal of machine learning research. 115 -143.

Gitari et al, Z. Z. (2015). “A lexicon-based approach for hate speech detection,” Int. J. Multimed. Ubiquitous Eng., vol. 10, no. 4, pp. 215–230, 2015.

Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2009). (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868.

Grosfoguel, R. (2016). What is racism?. . *Journal of World-Systems Research, 22(1),* , 9-15.

Guellil et al, I. S. (2021). . (2021). Arabic natural language processing: An overview. Journal of King Saud University-Computer and Information Sciences, 33(5), 497-507.

Guozhu Dong,Huan Liu . (2018). Feature Engineering for Machine Learning and Data Analytics.

H. Chen et al, S. M. (2017). “Abusive text detection using neural networks,” in CEUR Workshop Proceedings, 2017, vol. 2086,. pp. 258–260. .

Haidar et al, M. C. (2017). “A Multilingual System for Cyberbullying Detection : Arabic Content Detection using Machine Learning,” Adv. Sci. Technol. Eng. Syst. J., vol. 2, no. 6. pp. 275–284, .

Hasim Sak, A. W. (2014). “Long short-term memory recurrent neural network architectures for large scale acoustic modeling . 27.

Havaei et al, M. D.-F. (2017). Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., ... & Larochelle, H.. Brain tumor segmentation with deep neural networks.

Herz, M., & Molnár, P. (2012). *The content and context of hate speech: Rethinking regulation and responses.* Cambridge University Press.

Hochreiter& Schmidhuber . (1997). (1 November 1997). "Long Short-Term Memory". Neural Computation. 9 (8): 1735–1780. *Hochreiter, Sepp; Schmidhuber, Jürgen* , 1735-1780.

Horev, R. (2019). BERT explained: State of the art language model for NLP. https://bit. ly/2SYMYpy. Accessed November 16, 2019.

Horwitz, L. (2019). The future of IoT miniguide: The burgeoning IoT market continues Cisco. Source: https://www.cisco.com/c/en/us/solutions/internet-of-things/future-of-iot.html.

Hosseinmardi, H., Mattson, S. A., Rafiq, R. I., Han, R., Lv, Q., & Mishra, S. (2015). Analyzing labeled cyberbullying incidents on the instagram social network. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and lecture notes in bioinformatics . 49-66.

Howard, J. and Ruder, S. . (2018). Universal language model fine-tuning for text classification.CoRR, abs/1801.06146.

Husain, F. (2021). *ARABIC OFFENSIVE LANGUAGE DETECTION IN SOCIAL MEDIA.*

I. Kwok and Y. Wang. (2013). “Locate the Hate: Detecting Tweets against Blacks,” Twenty-Seventh AAAI Conf. Artif. Intell., pp. 1621–1622, 2013. .

Jacob Devlin, M.-W. C. (2019). BERT : Pre - training of deep bidirectional transformers for language under standing . In Proceedings of the 2019 Conference of the North American Chapter of the Association forComputational Linguistics : Human Language Technologies , Volume 1 (Long and . *(Long and Short Papers) , , Minneapolis , Minnesota . Associ ation for Computational Linguistics* . , pages 4171-4186 .

Joulin, A. G. (2016). Bag of Tricks for Efficient Text Classification. http://arxiv.org/abs/1607.01759.

Juvonen, J., & Graham, S. . (2014). 2014. *Bullying in schools: The power of bullies and the plight of victims. Annual review of psychology* , 65, 159-185.

Kalajdziski, S. (2018). ICT Innovations 2018. Engineering and Life Sciences. Cham:Springer. p. 220.

Krug E, Dahlberg L, Mercy J. (2002). *alWorld report on violence and health. Geneva: World Health Organization.* World Health Organization.

L. Kaati et al, E. O. (2016). “Detecting Multipliers of Jihadism on Twitter,” Proc. - 15th IEEE Int. Conf. Data Min. Work. ICDMW 2015, pp. 954–960, 2016.

LAURA J. LEDERER and RICHARD DELGADO (eds. (1995). The Price We Pay: The Case Against Racist Speech, Hate Propaganda, and Pornography.

Lecun, Yann, et al. (2015). Deep Learning.” Nature, vol. 521, no. 7553, 2015, doi:10.1038/nature14539. 436-444.

Lin et al, T.-Y. G. (2017). . Focal loss for dense object detection. Proceedings of the IEEE International Conference on Computer Vision, 2980–2988.

Loper, E., & Bird, S. (2002). Nltk: The natural language toolkit. arXiv preprint cs/0205028.

Luong, M.-T., Pham, H., & Manning, C. (2015). *Effective Approaches to Attention-based Neural Machine Translation.*

M. Di Capua et al, E. D. (2016). “Unsupervised cyber bullying detection in social networks,” in 2016 23rd International Conference on Pattern Recognition (ICPR), 2016. , pp. 432–437. .

Mikolov et al . (2013). Efficient estimation of word representations in vector space. arXivpreprint arXiv:1301.3781.

Mikolov, T. C. (2013). Efficient Estimation of Word Representations in Vector Space. http://arxiv.org/abs/1301.3781.

Mubarak, Hamdy, Kareem Darwish, and Walid Magdy. . (2017). . “Abusive Language Detection on Arabic Social Media,”. https://doi.org/10.18653/v1/w17-3008. 52-56.

N. Albadi et al, M. K. (2018). “Are they Our Brothers? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere,” 2018 IEEE/ACM Int. Conf. Adv. Soc. Networks Anal. .

N. Djuric et al, J. Z. (2015). “Hate Speech Detection with Comment Embeddings,” in Proceedings of the 24th International Conference on World Wide Web, 2015, . pp. 29–30.

Nahar et al, S. A.-M. (2014). “Semi-supervised Learning for Cyberbullying Detection in Social Networks,” in Databases Theory and Applications, 2014,. pp. 160–171. .

Nockleby, J. T. (2000). Hate speech. Encyclopedia of the American constitution, 3(2), 1277-1279. 1277-1279.

Obaidur Rahman, M., Sabir Hossain, M., Kamal Hossen, M., Junaid, T.-S., & Shafiul Alam Forhad, M. (2019). *Predicting Prices of Stock Market using Gated Recurrent Units (GRUs) Neural Networks Pattern Matching View project Algorithm View project Predicting Prices of Stock Market using Gated Recurrent Units (GRUs) Neural Networks.*

Özel et al, E. S. (2017). “Detection of cyberbullying on social media messages in Turkish,” in 2017 International Conference on Computer Science and Engineering (UBMK), 2017, . pp. 366–370. .

P. Badjatiya, S. G. (2017). “Deep Learning for Hate Speech Detection in Tweets,” in Proceedings of the 26th International Conference on World Wide Web Companion, 2017, .

P. Wadhwa and M. P. S. Bhatia. (2013). “Tracking on-line radicalization using investigative data mining,” in 2013 National Conference on Communications (NCC), 2013,. pp. 1–5. .

Paraschiv, A. . (2019). Upb at germeval-2019 task 2: Bert-based offensive language classification of german tweets. . *In Preliminary proceedings of the 15th Conference Erlangen, Germany: German Society for Computational Linguistics & Language Technology* , pages 396–402.

Park and Fung. (2017). “One-step and Two-step Classification for Abusive Language Detection on Twitter,” in AICS Conference,.

Pawar et al, Y. A. (2018). “Cyberbullying Detection System with Multiple Server Configurations,” 2018 IEEE Int. Conf. Electro/Information Technol., . pp. 90–95,.

Pedregosa F, V. G. (2011). Scikit-learn: Machine Learning in Python. JMLR. 12:2825–2830.

Peters, M. E. (2018). Deep contextualized word representations. CoRR, abs/1802.05365.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead Nature Machine Intelligence, 1 (5) . , pp. 206-215.

S. Agarwal and A. Sureka. (2015). “Using KNN and SVM Based One-Class Classifier for Detecting Online Radicalization on Twitter,” in International Conference on Distributed Computing and Internet Technology,. pp. 431–442.

S. Jaki and T. De Smedt. (2018). “Right-wing German Hate Speech on Twitter : Analysis and Automatic Detection,”. p29.

S. Malmasi and M. Zampieri. (2018). “Challenges in Discriminating Profanity from Hate Speech,” J. Exp. Theor. Artif. Intell., vol. 30, pp. 187–202, 2018. .

Saad. M. K. (2010). Arabic Morphological Tools forText Mining,‖ Int. Conf. Electr. Comput. Syst., pp. 1–6, 2010.

Schmidthuber, A. a. (2017). “A Survey on Hate Speech Detection Using Natural Language Processing,” no. 2012: 1–10. https://doi.org/10.18653/v1/w17-1101.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In 2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012, Kyoto Japan, March 25-30, 2012. pages 5149–5152.

Science, E. D. (2020). “What is Feature Engineering?”. Elite Data Science. Retrieved 10 February 2020 from: https://elitedatascience.com/feature-engineering.

Shekhar, A. (2018). (14 February 2018). What Is Feature Engineering for Machine Learning? Medium.com. Retrieved 10 February 2020 from: https://medium.com/mindorks/what-is-feature-engineering-for-machine-learning-d8ba3158d97a.

Shijie Wu and Mark Dredze. (2019). in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), , Hong Kong, China. Association for Computational Linguistics. pages 833–844.

Sinambela, E. (2010). A study of translating vulgarism by using euphemism. . *Jurnal Bahasa Dan Sastra "Langue",9, 2, 11–21* .

Springer, N., Engelmann, I., and Pfaffinger C. (s.d.). User comments: Motives and inhibitors t owrite and read. Information, Communication & Society, . 18(7):798–815.

Stone, G. R. (1994). Hate speech and the US Constitution. East European Constitutional Review. 3.78.

Straka, D. K. (2019). . 75 lan guages , 1 model : Parsing universal dependencies universally . In Proceedings of the 2019 Confer ence on Empirical Methods in Natural Language Processing and the 9th International Joint Confer ence on Natural. *Language Processing and the 9th International Joint Confer ence on Natural Language Processing (EMNLP IJCNLP) , Hong Kong , China . As sociation for Computational Linguistics* . pages 2779-2795.

Telmo Pires, Eva Schlinger, and Dan Garrette. (2019). How multilingual is multilingual BERT? In Proceedings of the 57th Annual Meeting of the Asso ciation for Computational Linguistics, Florence, Italy. Association for Computational Linguistics. , pages 4996– 5001.

TORRES.AI, J. (2018). Learning process of a neural network How Do Artificial Neural Networks Learn? Sep 21 2018 https://towardsdatascience.com/how-do-artificial-neuralnetworks-learn-773e46399fc7.

Volokh & Eugene. (2015). The speech integral to criminal conduct exception. Cornell L. Rev. 101, 981.

W. Warner and J. Hirschberg. (2012). "Detecting Hate Speech on the World Wide Web," no. Lsm, pp. 19–26, .

Wadhwa & Bhatia. (2013). "Tracking on-line radicalization using investigative data mining," in 2013 National Conference on Communications (NCC), 2013, pp. 1–5. .

Wang, Y., Liao, W., & Chang, Y. (2018). Gated Recurrent Unit Network-Based Short-Term Photovoltaic Forecasting. Energies 2018, 11, 2163. .

Waseem, Z., & Hovy, D. (2016). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In NAACL-HLT . . *https://aclanthology.org/N16-2013/* , (pp. 88–93). .

Watanabe et al, M. B. (2018.). "Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection," IEEE Access, vol. 6,.

WHO, World Health Organisation. (2004). preventing violence A: guide to implementing the recommendation of the world Report on violence and health.

Wiegand et al, J. R. (2018). “Inducing a Lexicon of Abusive Words – a Feature-Based Approach,” in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (LongPapers), 2018, . pp. 1046–105.

Wilson, A. M. (1982). A familiar face. In Anaesthesia (Vol. 37, Issue 12). https://doi.org/10.1111/j.1365-2044.1982.tb01804.x.

Xu, K., Lei Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., et al. (2015). *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention.*

Y. Chen et al, Y. Z. (2012). “Detecting offensive language in social media to protect adolescent online safety,” in Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust and 2012 ASE/IEEE International Conference on Social Computing . *SocialCom/PASSAT 2012, 2012, no. December,* , pp. 71–80. .

Yang et al, Y. Z. (2018). Glomo Unsupervisedly learned relational graphs as transferable representations. ArXiv Preprint ArXiv1806.05662.

Z. Waseem and D. Hovy. (2016). hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter,” Proc. NAACL Student Res. Work., . pp. 88–93.

Z. Zhang and L. Luo. (2018). “Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter,” vol. 1, no. 0, pp. 1–5, 2018.

Zhang, Y. and Wallace, B. C. (2015). A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification. CoRR, abs/1510.03820.