Democratic and Popular Republic of Algeria
Ministry of higher education and scientific research
Faculty of science
Department of mathematics and computer science

Graduation Project

Submitted for graduation in
Computer science master
Specialty: System Engineering and Web Technology

AUTOMATIC DETECTION OF HATE SPEECH ON SOCIAL MEDIA FOR THE ALGERIAN DIALECT

Proposed and directed by:
M. Mazari Ahmed Cherif

Prepared and presented by:
Ms.Chiker Nourelhouda Ms.Ben khaoua Sarah

2020/2021

Acknowledgment:

First of all, thanks to ALLAH for giving us the strength and ability to complete this project.

Then we would like to thank our supervisor, M. MAZARI Ahmed Cherif, who has given us the opportunity to work on this project and has provided us with all the support and leadership.

Then, a special thanks to all the amazing people on Github, Stackoverflow, and Kaggel who had answered all our questions and provided us with a lot of information.

And also, all the professors at our university who helped us during our studies.

Last but not least, we would like to thank our families and our friends for their support and encouragement.

Dedication

This thesis is dedicated to:

The sake of Allah, my Creator and my Master,

My great parents, who never stop giving of themselves in countless ways,

My beloved brothers and sisters, who stands by me when things look bleak,

My friends who encourage and support me,

All the people in my life who touch my heart, I dedicate this research

NOUR EL HOUDA

Dedication

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

To My parents, for their understanding and support morally and financially,

To My lovely brothers and sisters, my eternal gratitude

To all my friends.

SARAH

Abstract

Hate speech on social media platforms can lead to cyber-crime, affecting social life on a personal and national level. Hateful speech spread through social media has the potential to cause personal harm and suffering, as well as social tension. Social media platforms, on the other hand, are unable to regulate all of the contents that users post. As a result, there is demand for automatic detection of hate speech. This need is increased when the posts are written in complex languages, such as Arabic, which is known with its challenges, difficulties and lack of resources. In this work we focused on the Algerian dialects. We release a manually annotated multi-label Algerian hate speech detection dataset. Our dataset contains 14k comments extracted from Facebook, YouTube and Twitter. To confirm our annotated dataset's practical utility, we used it to train and evaluate several classification models started with the machine leaning classification models, followed by the basic deep learning models, then we test several BERT based models built by either adding DL layers or by composing ensemble learning models constituted of BERT model and other DL models trained individually and finally reported the obtained results, as a best results we got 74% accuracy and 76% F1-Score with ensemble learning (Bert with GRU,LSTM and both of GRU and LSTM). Though the dataset is mainly designed for hate speech detection, it can also be used for cyberbullying detection, and offensive and abusive language detection.

Keywords: hate speech, social media, cyberbullying, offensive and abusive language, deep learning, machine learning, transfer learning, ensemble learning.

Résume

Les discours de haine sur les réseaux sociaux peuvent conduire à des crimes cyber, affectant la vie sociale au niveau personnel et national, Les discours haineux diffusés sur les réseaux sociaux peuvent potentiellement causer des préjudices et des souffrances personnelles, Ainsi que les tensions sociales, D'autre part, ces plateformes ne sont pas capables de réguler tous les contenus que les utilisateurs publient. En conséquence, il y a une demande de trouver une détection automatique pour les discours de haine, Cette demande est accrue lorsque les postes sont rédigés dans des langues complexes comme l'arabe qui est connu avec ses défis, ses difficultés et son manque de ressources. Dans ce travail, on a concentré sur les dialectes algériens, nous publions une base de données annotés manuellement de détection des discours de haine algériens multi-étiquettes. Notre base de données contient 14K commentaires extraits de Facebook YouTube et Twitter. pour confirmer l'utilité pratique de notre base de données annoté, nous l'avons utilisé pour former et évaluer plusieurs modèles de classification commencés avec le modèle de classification d'apprentissage automatique, Suivis par les modèles de base d'apprentissage profond, puis on a testé plusieurs modèles basés sur BERT construits entières par des couches DL ou en composant des modèles d'apprentissage d'ensemble constitués du modèle BERT et d'autres modèles DL Formé individuellement et finalement rapporté les résultats obtenus on a eu 74% accuracy et 76% F1-Score comme une meilleur résultat avec l'apprentissage par ensemble (Bert, GRU, Lstm et LSTM avec GRU) , Bien que cette base de données est designer principalement pour la détection des discours de haine, elle peut également être utilisé pour la détection de la cyber harcèlement et la détection d'un langage offensant ou abusif

Mots clé : Discours de haine, réseaux sociaux, cyber harcèlement, discours offensant et abusif, apprentissage en profondeur, apprentissage automatique, apprentissage par transfert, apprentissage par ensemble.

ملخص

خطابات الكراهية في مواقع التواصل الاجتماعي يمكنها ان تؤدي الى جرائم الكترونية لسيما تؤثر سلبيا على حياة الفرد و المجتمع .و لقد انتشرت بكثرة على وسائل التواصل الاجتماعي و لها القدرة على التسبب في الاضرار و معانات الاشخاص وتستطيع ايضا ان تؤدي الى توتر اجتماعي . وسائل التواصل الاجتماعي ليس لها القدرة على التحكم على المنشورات و من اجل هذا نسعى للبحث بطريقة الية على هذه الخطابات، و هذه الحاجة تزداد كلما زادت المنشورات المكتوبة بلغة معقدة كالعربية كما نعلم انها لغة صعبة و لها نقص في المصادر. في هذه الاطروحة نعمل على اللهجة الجزائرية . عملنا قاعدة بيانات خاصة بالبحث عن خطاب الكراهية تحتوي هذه البيانات على 14 الاف تعليق من اليوتيوب ، فايسبوك و تويتر لتاكيدها . استعملنا تدريب و تقييم العديد من نماذج التصنيف . بدائنا بالتعليم الالي ثم بنماذج التعليم العميق الاساسية، ثم قومنا باختبار العديد من النماذج المعتمدة على BERT التي نشأنها مرة عن طريق اضافة طبقات التعلم العميق و مرة بطريقة تجميع النماذج التي قمنا بتدريبها على عمل بشكل فردي و اخيرا عرض النتائج المحصلة عليها حيث تحصلنا على % 74 في accuracy و76% في F1-Score كأحسن نتيجة بأستعمال التعليم بالتجميع (LSTM, Gru, BERT) . برغم من ان هته البيانات مخصصة لخطابات الكراهية الى انها يمكن استعلمها للكشف عن تنمر الاكتروني و كذلك الكلام الجارح و المسيئ

الكلمات المفتاحية: خطاب الكراهية - مواقع التواصل - التنمر الاكتروني - الكلام الجارح و المسيئ - التعليم العميق - التعليم الالي - التعليم التحويلي – التعليم بالتجميع

Table of Contents

List of figures I

List of tables III

List of acronyms IV

General introduction 1

Chapter 01: Hate speech 3

Introduction 3

1. Definition and terminologies 3

1.1 Hate speech 3

1.2 Hate Speech and Other Related Concepts: 5

1.2.1 Offensive language: 5

1.2.2 Abusive language: 6

1.2.3 Cyber hate: 6

1.2.4 Cyberbullying: 6

1.3 Comparison between Hate Speech Definition and Related Concepts: 7

2. Impact of hate speech: 8

3. Against hate speech: 8

4. Online hate speech and the Arabic language: 10

4.1 Arabic in social media: 10

4.2 Arabic dialect problems challenges: 11

5. Proposed solution: 11

5.1 The need of automatic hate speech detection: 11

5.1.1 Automated Techniques for Hate Speech Detection: 12

6. Challenges in Hate Speech Detection: 14

Conclusion: 14

Chapter 02: Deep learning and transfer learning 15

Part1: 15

Introduction 15

1. Deep learning 15

1.1. Artificial Intelligence, Machine Learning and Deep Learning 15

1.2. Feature Learning 16

1.3 Types of deep learning approaches 16

1.3.1 Supervised learning 16

1.3.2. Unsupervised learning 16

1.3.3. Semi supervised learning 16

1.3.4. Deep Reinforcement Learning (RL) 17
1.4. Why and when to apply DL 17
1.5. Challenges of Deep learning 18
2. Deep neural network process 19
2.5. History 19
2.6. Gradient Descent 19
2.7. Stochastic Gradient Descent (SGD) 20
2.8. Back-Propagation (BP) 20
2.9. Activation fonction 20
2.9.1. Linear: 20
2.9.2. Sigmoid: 20
2.9.3. Tanh: 20
2.9.4. SoftMax: 20
2.9.5. ReLU: 21
2.10. Cost function 21
2.11. Regularization 22
2.11.1. Dropout 22
2.11.2. Drop connect 23
3. Convolutional Neural Network (CNN) 23
3.5. CNN Overview 23
3.6. Convolution Layer 23
3.7. Sub-sampling Layer 24
3.8. Classification Layer 24
4. Transfer learning 24
4.5. What Is a Pre-trained Model? 25
4.6. Why Will You Use Pre-Trained Models? 25
4.7. How Will You Use Pre-Trained Models? 25
5. Recurrent Neural Network (RNN) 26
6. Long Short-Term Memory (LSTM) 28
7. Gated Recurrent Unit (Gru) 29
8. Attention 30
8.5. Attention Mechanism 30
8.5.1. Self Attention 32
8.5.2. Global attention 32
8.5.3. Local attention 32

8.6. Transformer: 32
8.7. Auto encoder (AE) 34
9. Embedded Vectors: 36
9.5. Collobert and Weston (C&W) model: 36
9.6. Word2vec: 36
9.7. CBOW: 37
9.8. Skip gram: 37
9.9. FastText: 37
9.10. Global Vectors for Word Representation (GloVe): 38
9.11. Embeddings from Language Models (ELMo): 38
9.11.1. ELMo Architecture: 39
9.12. Graphs from LOw-level unit Modeling (GLoMo): 39
9.13. Universal Language Model Fine-tuning (ULMFiT): 40
9.14. Bidirectional BERTEncoder Representation from Transformers (BERT): 40
9.14.1. Model Architecture: 40
9.14.2. BERT Input Representation: 41
9.14.3. BERT Output Representation: 42
Part2: 43
1. Related work: 43
1.1. Current state in Hate Speech Detection and Related Concepts: 43
1.2. Abusive language detection: 43
1.3. Cyber bullying detection: 43
1.4. Hate speech detection: 44
1.5. Arabic Hate Speech Detection: 44
1.5.1. Arabic anti-social behavior detection: 45
1.5.2. Arabic hate speech detection: 46
1.5.3. Algerian dialect hate speech detection: 46
1.6. Summary and Analysis 47
Conclusion 56
Chapter 03: Proposed approach for Algerian hate speech detection 57
Introduction 57
1. Global Architecture 57
2. Data creation: 59
2.1. Data collection: 59
2.2. Retrieving methods: 59

2.2.1. Facebook API (application programming interface) 59
2.2.2. Twitter API (application programming interface) 60
2.2.3. YouTube API (application programming interface) 60
2.3. Saving comments 60
2.4. Comments filtering 60
2.5. Comments annotation 61
3. Data preprocessing 61
3.1. Cleaning comments 61
3.2. For the deep learning basic models 62
3.2.1. Tokenization 62
3.2.2. pad sequence to a specific length 62
3.3. For transfer learning models 62
3.3.1. Tokenization: 62
3.3.2. pad sequence to a specific length 62
3.3.3. Creating attention masks for sentences tokens 62
4. Features extraction 63
5. Classification 63
5.1. Machine learning 63
5.2. deep learning 63
5.2.1. LSTM, GRU, Bi-LSTM and Bi-GRU 63
5.2.2. CNN 64
5.3. Transfer learning 65
5.3.1. Bert model 65
5.3.1.1. Bert embedding +LSTM 68
5.3.1.2. Bert embedding + GRU 68
5.3.1.3. Bert embedding and GRU+LSTM 69
5.4. Ensemble learning 70
6. Evaluation Measures 73
Conclusion 73
Chapter 04: Test and results 70
Introduction 74
1. Environment and developments tools 74
1.1. Google Colaboratory: 74
1.2. Data extraction: 74
1.2.1. Python 74

1.2.2. Json 75
1.2.3. Api client 75
1.3. Preprocessing 75
1.3.1. Pandas 75
1.3.2. NLTK 75
1.3.3. NumPy 75
1.3.4. Regular expression 76
1.3.5. Remove other language letters 76
1.4. Fine-tuning pretrained language models 76
1.4.1. Transformers Hugging Face: 76
1.4.2. Tensorflow 76
1.4.3. BertTokenizer 76
1.4.4. DistilBertTokenizer 77
1.4.5. Arabert Tokenizer 77
1.4.6. Bert for Sequence Classification 77
1.4.7. DistilBert for Sequence Classification 77
1.4.8. AraBERT for sequence classification 77
1.5. Training deep learning architecture 77
1.5.1. Keras 77
1.5.2. Genism 78
1.6. Performance Evaluation Metrics 78
2. Dataset composition 78
2.1. Our dataset 79
2.2. The second dataset 81
3. Test and evaluation 82
3.1. Training and Test Strategies 82
3.1.1. Hyper parameterization 82
3.1.2. Preprocessing for models 82
4. Test results for models 85
4.1. Machine learning models 85
4.1.1. With TF-IDF embedding 86
4.1.2. With Word2Vec embedding 87
4.2. Deep learning models 85
4.3. For transfer learning: 87
4.3.1. Bert layer with GRU and LSTM 92

4.4. Ensemble learning 95
4.5. Comparison between deep learning models and pretrained language models 97
4.6. General accuracy comparison..... 99
4.7. General f1_score comparison 100
Conclusion..... 103
Conclusion and perspective 105
Bibliography

List of figures

Figure 1: Hate speech statistics 5
Figure 2: Hate speech laws around the world 10
Figure 3: Performance of deep learning with respect to the amount of data 18
Figure 4: Pictorial representation of the concept Dropout. 22
Figure 5: Conceptual diagram for transfer learning: Pretrained on ImageNet and transfer learning is used for retraining on PASCAL dataset. 25
Figure 6: Structure of basic Recurrent Neural Network (RNN) with a loop. 26
Figure 7: Different structure of RNN with respect to the applications: (a) One to one; (b) many to one; (c) one to many; (d) many to many; and (e) many to many 27
Figure 8: Unrolled RNNs 28
Figure 9: Diagram for Long Short-Term Memory (LSTM). 29
Figure 10 : Diagram for Gated Recurrent Unit (GRU) 30
Figure 11: Sequence to Sequence Model architecture using Attention Mechanism. (Bahdanau et al., 2015) 31
Figure 12: Global and local attention model 32
Figure 13: Transformer model architecture. (Vaswani et al., 2017) 34
Figure 14: Diagram for Auto encoder 35
Figure 15: Skip gram and CBOW model 37
Figure 16: ELMo Architecture. (Peters et al., 2018) 39
Figure 17: Architecture from BERT (Devlin et al., 2019) 41
Figure 18: BERT input representation 42
Figure 19: BERT outputs representation 42
Figure 20: General steps for constructing the annotated dataset 57
Figure 21: DL and ML classification 58
Figure 22: Global overview architecture 58
Figure 23: Steps of data collection and preprocessing. 59
Figure 24: Information about the collected dataset. 61
Figure 25: DL architecture 64
Figure 26: CNN architecture 65
Figure 27: BERT architecture 67
Figure 28: BERT and DL layer architecture 67
Figure 29: Bert layer with LSTM layer 68
Figure 30: Bert layer with GRU 69
Figure 31: Bert layer with both GRU and LSTM 70
Figure 32: Bert and LSTM with Bert embedding 70
Figure 33: Bert and GRU with Bert embedding 71
Figure 34: Bert and both GRU and LSTM with Bert embedding 71
Figure 35*:* Ensemble learning architecture 72
Figure 36: Process of comparing between the two datasets 78
Figure 37: Dataset topics 79
Figure 38: Dataset annotation 80
Figure 39: Percentage of cyberbullying 80
Figure 40: Percentage Hate speech 80
Figure 41: Percentage of abusive and offensive language 80
Figure 42: Dataset annotation 81
Figure 43: Bert with LSTM architecture 93
Figure 44: Multitopic and misogyny datasets comparison between LSTM and GRU with Bert layer 95
Figure 45: Multitopic and misogyny dataset accuracy, f1-score comparison for ensemble learning... 97
Figure 46: Deep learning / pretrained language models Comparison 98

Figure 47: General accuracy comparison 100
Figure 48 : General f1_score comparison 102

List of tables

Table 1: Comparison between Hate Speech Definition and Related Concepts 7
Table 2: Criterions need to be considered for transfer learning. 26
Table 3: Summary of the current state of anti-social behaviour detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F). 48
Table 4: Summary of the current state of cyberbullying detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F). 49
Table 5: Summary of the current state of religion hate speech detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (F). 50
Table 6: Summary of the Italian,Indonesia and German contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F). 52
Table 7: Summary of the Arabic contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (F). 53
Table 8: Summary of the Algerian dialect contributions in hate speech detection and their respective results, in the metrics: Precision (P), Recall (R), F1-Score (F). 56
Table 9: Information about the dataset 61
Table 10: Information about the collected data. 79
Table 11: Dataset polarity 81
Table 12: Results of TF-IDF with ML 86
Table 13: Test results for SKIP-Gram and SBOW with ML 84
Table 14: Results of ensemble models. 96
Table 15: Deep learning / pretrained language models Comparison 98
Table 16: General accuracy comparison between all the models 99
Table 17: General f1_score comparison between all the models 100

List of acronyms

AI: Artificial Intelligence.
AMT: Mechanical Turk.
ANN: Artificial Neural Networks.
API: Application Programming Interface.
AUC: Area Under the Curve.
BERT: Bidirectional Encoder Representation from Transformers.
Bi-GRU: Bidirectional Gated Recurrent Unit.
biLMs: Bidirectional Language Models.
Bi-LSTM: Bidirectional Long Short-Term Memory.
Bi-RNN: Bidirectional Recurrent Neural Networks.
BPE: Byte Pair Encoding.
C&W: Collobert and Weston.
CBOW: Continuous Bag of Words.
CF: CrowdFlower.
CNN: Convolutional Neural Networks.
CPU: Central Processing Unit.
DistilBERT: Distilled Version of BERT.
DL: deep learning
DNN: Deep Neural Network.
EL: ensemble learning
ELMo: Embeddings from Language Models.
FNN: Feed Forward Neural Network.
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
NL: Normalization Layer.
NLP: Natural Language Processing.
OLID: Offensive Language Identification Dataset.
OOV: Out of Vocabulary.
POS: Part of Speech.
RF: Random Forest.
RNN: Recurrent Neural Networks.
RoBERTa: Robustly Optimized BERT Approach.
ROC: Receiver Operating Characteristics.
SNS: Social Networking Services.
SVC: Support Vector Classifier.
SGD: Stochastic Gradient Descent.
TF: transfer learning
TF-IDF: Term frequency-inverse document frequency
TPU: Tensor Processing Unit.
TRAC: Trolling, Aggression, and Cyberbullying.
ULMFiT: Universal Language Model Fine-tuning.
URL: Uniform Resource Locator.
W2V: Word To Vector

General introduction

General introduction

Over the last decades, people are getting more engaged with the wide spread of Web2.0. Social networks opened up the chance for people around the globe to express and share their thoughts extensively and in a real-time manner. Such expressions afford researchers with the ability to investigate the online social emotions in different events. people now have the potential to speak freely, this allowed them to exchange all sorts of thoughts, emotions and knowledge. However, cyberspace is not always safe, it can be a reason for the dissemination of aggressive and harmful content.

Hate speech is an online common form for expressing prejudice and aggression. This may convey racist, xenophobic and many forms of verbal aggression. Hate speech is typically defined as the act that disparages a person or people on the basis of a number of characteristics that may include and not limited to: race, ethnicity, sexual orientation, gender, religion and nationality. (Schmidt & Wiegand, 2017)

In social media platforms, there are uncontrollable number of comments and posts issued every second which make it impossible to trace or control the content of such platform. Therefore, social platforms are facing a problem in limiting these posts while balancing the freedom of speech. In addition, the diversity of people and their backgrounds, cultures and believes can ignite the flame of hate speech. In the other hand, each culture has its own different interpretations and characteristics of cyber-hate. So, every culture is assumed to act differently and have their own way of intervention in a manner which best suits the culture.

For the Arab region, there is a noticeable growth in the usage of social media platforms. According to the Arab social media report[1], the social media penetration in the Arab region reached 90% of the population in some countries. and as it is knowing hate speech detection in Arabic in more difficult due to the complexity of the language in terms of morphology, spelling, and dialects. In this study, we concentrate on Algerian dialect, which is spoken by the majority of the Algerian population. in which this dialect imposes additional challenges such as the combination of standard Arabic, standard French, and additional terms unique to the Algerian dialect. Because of its complicated nature, few research have focused on Algerian dialect.

this increase in usage and the openness in speech results in a public concern on existing practices in social networks. A vast number of posts that is hard and even impossible to control manually by platform owners. Hate and aggression through social networks should be rationed and regulated by policy makers and should be also countered by harnessing the power of artificial intelligence and machine learning algorithms to automate the detection of hate speech in social media.

A lot of research studies have been presented in the last few years, only a few of these studies are dedicated to collect Algerian dialect's dataset and detect hate speech. However, several elements, such as dataset processing and the selection of appropriate classification models, are lacking. therefore, we propose to create Algerian hate speech detection dataset, then applicate the basic classification models to confirm our data and then proposing a transfer learning model.

This report contains four sections:

- **Chapter 01: Hate speech**

In this chapter we defined hate speech and other related concepts, such as cyberbullying, abusive language, etc., and the difference between them as well as their impact on individuals and communities. Then we present online hate speech for the Arabic language and its

[1] Arab-Social-Media-Report.pdf (interactiveme.com) available at: https://interactiveme.com/wp-content/uploads/2011/02

General introduction

challenges, especially the Algerian dialect. Finally, we discussed the current automated technique for hate speech detection and its difficulties.

- **Chapter 02: Deep learning and transfer learning**

In this section, first we illustrate the different deep learning approaches, furthermore transfer learning and ensemble learning. Second, we discussed some relevant work in both English and Arabic, as well as the Algerian dialect.

- **Chapter03: Proposed approach for Algerian hate speech detection**

In this third section, we present the main proposed approaches to deal with hate speech detection for Algerian dialect, which are machine learning classifiers, deep learning classifiers, fine tuning pretrained language models and creating a deep learning architecture.

- **Chapter 04: Test and results**

In this chapter, we begin by describing the environment and the many modules we use to train, finetune, and test our models. After that, we show the many procedures we take to create our dataset, followed by test results for each model.

CHAPTER 01:
HATE SPEECH

Table of Contents

Chapter 01: HATE SPEECH 3
Introduction 3
1. Definition and terminologies 3
1.1 Hate speech 3
1.2 Hate Speech and Other Related Concepts: 5
1.2.1 Offensive language: 5
1.2.2 Abusive language: 6
1.2.3 Cyber hate: 6
1.2.4 Cyberbullying: 6
1.3 Comparison between Hate Speech Definition and Related Concepts: 7
2. Impact of hate speech: 8
3. Against hate speech: 8
4. Online hate speech and the Arabic language: 10
4.1 Arabic in social media: 10
4.2 Arabic dialect problems challenges: 11
5. Proposed solution: 11
5.1 The need of automatic hate speech detection: 11
5.1.1 Automated Techniques for Hate Speech Detection: 12
6. Challenges in Hate Speech Detection: 14
Conclusion: 14

Hate speech

Introduction

Nowadays, the spread of hate speech has become an uncontrollable problem due to its online appearance, which causes harmful effects on individuals and communities and can also lead to illegal activities.

In this chapter, we will define hate speech and its various categories, as well as its effects, online hate speech and the Algerian dialect.

1. Definition and terminologies

1.1 Hate speech

Deciding if an expression includes hate speech or not is not simple, even for humans. Hate speech is a complex problem that is linked to group relationships and often relies on language nuances. As a result, it's important to characterize hate speech precisely in order to make automated detection of it easier. We gathered different definitions of hate speech and compared viewpoints from a variety of outlets.

- **European Union Commission**:

"All conduct publicly inciting to violence or hatred directed against group of persons or a member of such a group defined by reference companies to race, color, religion, descent or national or ethnic" (Jourová, 2017)

- **International minorities associations (ILGA):**

"Hate speech is public expressions which spread, incite, promote or justify hatred, discrimination or hostility toward a specific group. They contribute to a general climate of intolerance which in turn makes attacks more probable against those given groups." (ILGA, 2016a)

- **Nobata et al**:

"Language which attacks or demeans a group based on race, ethnic origin, religion, disability, gender, age, disability, or sexual orientation/gender identity." (Nobata et al., 2016a)

Hate speech

- **Facebook**:

"Content that attacks people based on their actual or perceived race, ethnicity, national origin, religion, sex, gender or gender identity, sexual orientation, disability or disease is not allowed. We do, however, allow clear attempts at humor or satire that might otherwise be considered a possible threat or attack. This includes content that many people may find to be in bad taste (ex: jokes, stand-up comedy, popular song lyrics, etc.)." [2]

- **YouTube**:

"Hate speech refers to content that promotes violence or hatred against individuals or groups based on certain attributes, such as race or ethnic origin, religion, disability, gender, age, veteran status and sexual orientation/gender identity. There is a fine line between what is and what is not considered to be hate speech. For instance, it is generally okay to criticize a nation-state, but not okay to post malicious hateful comments about a group of people solely based on their ethnicity." [3]

- **Twitter:**

"Hateful conduct: You may not promote violence against or directly attack or threaten other people on the basis of race, ethnicity, national origin, sexual orientation, gender, gender identity, religious affiliation, age, disability, or disease." [4]

So, we can define hate speech as any offensive, abusive, insulting language against a person or group (Faris et al., 2020). According to (Chetty & Alathur, 2018a) hate speech is expression that conduct to violence based on some characteristic such as ethnicity, nationality, religion, sexual orientation and gender identity. Therefore, it can be classified into the following categories: Figure 1 shows hate speech categories statistics.

1.1.1 Religion hate speech

This category contains any kind of hatred expression toward religions such as: Islam, Hindu, and Christian…etc. It is one of the most common motives for crimes. (Albadi et al., 2018a)

1.1.2 Gendered hate speech

This category includes any offensive expressions or devaluation of a person's gender, as well as any form of misogyny. Generally, this type of hate speech targets women and girls. (Chetty & Alathur, 2018b)

[2] Facebook. 2013 https://www.facebook.com/help/%20135402139904490
[3] Youtube. 2017. https://support.google.com/youtube/answer/2801939?hl=en
[4] Twitter. 2017 https://support.twitter.com/articles/

1.1.3 Racist hate speech

This type contains any kind of expression directed against a person on the basis of their physical appearance like skin color or ethnicity, nationality…etc. (Chetty & Alathur, 2018b)

1.1.4 disability hate speech

Including any sort of offence to an individual suffering from health which limits to do some of the life activities.

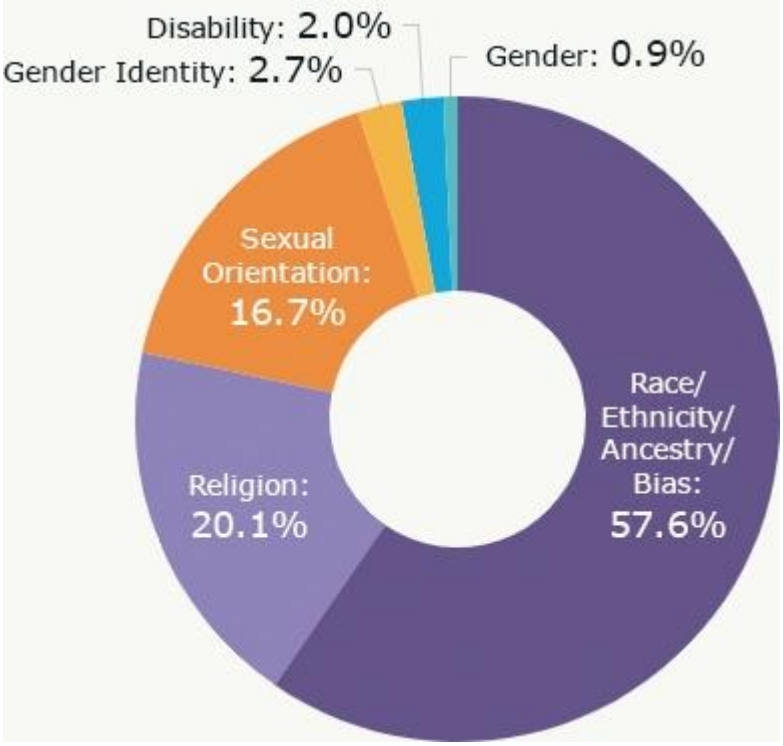


Figure 1: Hate speech statistics

1.2 Hate Speech and Other Related Concepts:

In the previous sections, we analyzed different definitions of hate speech. Another way of better understanding this complex phenomenon is by comparison with other related concepts.

1.2.1 Offensive language:

Is a phenomenon that is closely linked to a variety of other linguistic and cultural phenomena, such as violent and aggressive words, cyberbullying, bigotry, extremism, radicalization, toxicity, profanity, arson, sexism, hatred, and hate speech. (Oxford English Dictionary)

Hate speech

1.2.2 Abusive language:

In common English dictionaries, abusive language is described as "highly offensive and insulting; engaged in or marked by habitual violence and cruelty." (The Oxford English Dictionary, 2019), while (Fortuna & Nunes, 2018) characterize abusive vocabulary as follows: “any strongly impolite, rude or hurtful language using profanity, that can show a debasement of someone or something, or show intense emotion.”

1.2.3 Cyber hate:

Hate speech is an old problem that has found a new way to be easily expressed on a variety of platforms such as Facebook, Twitter, YouTube where there are a huge number of publications that can be created each second, making it difficult to manually distinguish between hate speech and permissible speech. Online hate speech or Cyber-hate is described as "the use of electronic communications technologies to disseminate racial, ethnic, militant, or terrorist messages." It differs from cyber-bullying in that hate speech can affect whole groups (C.Blaya 2018).

1.2.4 Cyberbullying:

Is generally characterized as insults or threats directed at a specific person (Zampieri et al., 2019a). Purpose to damage, repetitiveness, and an inequality of power are all listed as signs of cyberbullying (Van Hee et al., 2015). This form of online abuse is most popular among children and teens, and cyberbullying is illegal in a number of countries, as well as several states in the United States (Gregorie, 2001).

1.3 Comparison between Hate Speech Definition and Related Concepts:

Table 1: Comparison between Hate Speech Definition and Related Concepts

Concept	Definition of the concept	Distinction from hate speech
Hate	Expression of hostility without any stated explanation for it (Tarasova, 2016).	Hate speech is hate focused on stereotypes, and not so general.
Cyberbullying	Aggressive and intentional act carried out by a group or individual, using electronic forms of contact, repeatedly and over time, against a victim who cannot easily defend him or herself (Chen et al., 2012).	Hate speech is more general and not necessarily focused on a specific person.
Abusive language	The term abusive language was used to refer to hurtful language and includes hate speech, derogatory language and also profanity (Nobata et al., 2016b).	Hate speech is a type of abusive language.
Toxic language or comment	Toxic comments are rude, disrespectful or unreasonable messages that are likely to make a person to leave a discussion (Hosseini et al., 2017).	Not all toxic comments contain hate speech. Also, some hate speech can make people discuss more.
Profanity	Offensive or obscene word or phrase.[5]	Hate speech can use profanity, but not necessarily.

[5] PROFANITY | meaning in the Cambridge English Dictionary

Extremism	Ideology associated with extremists or hate groups, promoting violence, often aiming to segment populations and reclaiming status, where outgroups are presented both as perpetrators or inferior populations (McNamee et al., 2010a).	Extremist discourses use frequently hate speech. However, these discourses focus other topics as well, (McNamee et al., 2010b).
Radicalization	Online radicalization is similar to the extremism concept and has been studied on multiple topics and domains, such as terrorism, anti-black communities, or nationalism (Agarwal & Sureka, 2015).	Radical discourses, like extremism, can use hate speech. However, in radical discourses topics like war, religion and negative emotions (Agarwal & Sureka, 2015). are common while hate speech can be more subtle and grounded in stereotypes.

2. Impact of hate speech:

Hate motivated speech is considered an offense in many countries around the world, we know that hate has a propensity to spread expansively and rapidly, and even motivate individuals who are inclined towards hate, furthermore it could lead to violence or a serious crime in real life such as the "genocide of the Rohingya community in Myanmar, anti-Muslim mob violence in Sri Lanka, and the Pittsburg shooting" (Mathew et al., 2020), It also has psychological effects, such as depression, anxiety, poor self-worth, and, in some situations, suicide.

3. Against hate speech:

The" United Nation Strategy and Plan of Action on Hate Speech" was officially released by the United Nations which seeks to enhance UN efforts to fight hate speech's sources. Data management and analysis, using technology, and engagement with digital and conventional media are all part of the strategy. It also calls for further study into the connection between the abuse of the Internet and social media for the dissemination of hate speech and the causes that lead people to violence.[6]

[6] https://www.un.org/en/genocideprevention/documents/UNStrategyandPlanofActiononHateSpeech2018JuneSYNOPSIS

Hate speech

The European commission against racism and intolerance calls for speedy reactions by public figures to hate speech, promotion of self-regulation of media, raising awareness of the dangerous consequences of hate speech, withdrawing financial and other support from political parties that actively use hate speech, and criminalizing its most extreme manifestations, while respecting freedom of expression.[7]

There was no formal or clear law banning hate speech in the Algerian Penal Code, until 28th of April, 2020 the law No.20/05, which prohibits and combats bigotry and hate speech define hate speech as any kind of expression that spreads, incites, encourages, or justifies racial hatred, and other form of hatred such as humiliation, hostility, or violence against person or group on the basis of their race, religion, ethnic origin, national origin, sex, Health status, disability, Geographical affiliation, or gender identity.

It does, however, contain a ban against anyone that produces, maintains, or supervises a dedicated website or electronic account for the purpose of endorsing any initiative, concept, news, or sketches or pictures that may incite bigotry and hate in society which can result in a sentence of five to ten years in prison and a penalty of 5,000,000 to 10,000,000 Dinars.[8]

There is also a prohibition against insulting Islam that carries three to five years in prison and/or a fine of 50,000 to 100,000 Dinars, Insult is described as insulting the Prophet or any of the Islamic rituals in any way, whether by poetry, drawing, comments, or some other means.[9]

Hate speech and posts encouraging abuse or hatred towards people was prohibited on social media platforms such as Facebook, Twitter, and YouTube, according to their terms and policies.

[7] ECRI General Policy Recommendation N°15 (coe.int)
[8] Article 34 of the Algerian Constitution
[9] Article 44 of the Algerian Penal Law. Available at: http://www.joradp.dz/TRV/APenal.pdf.

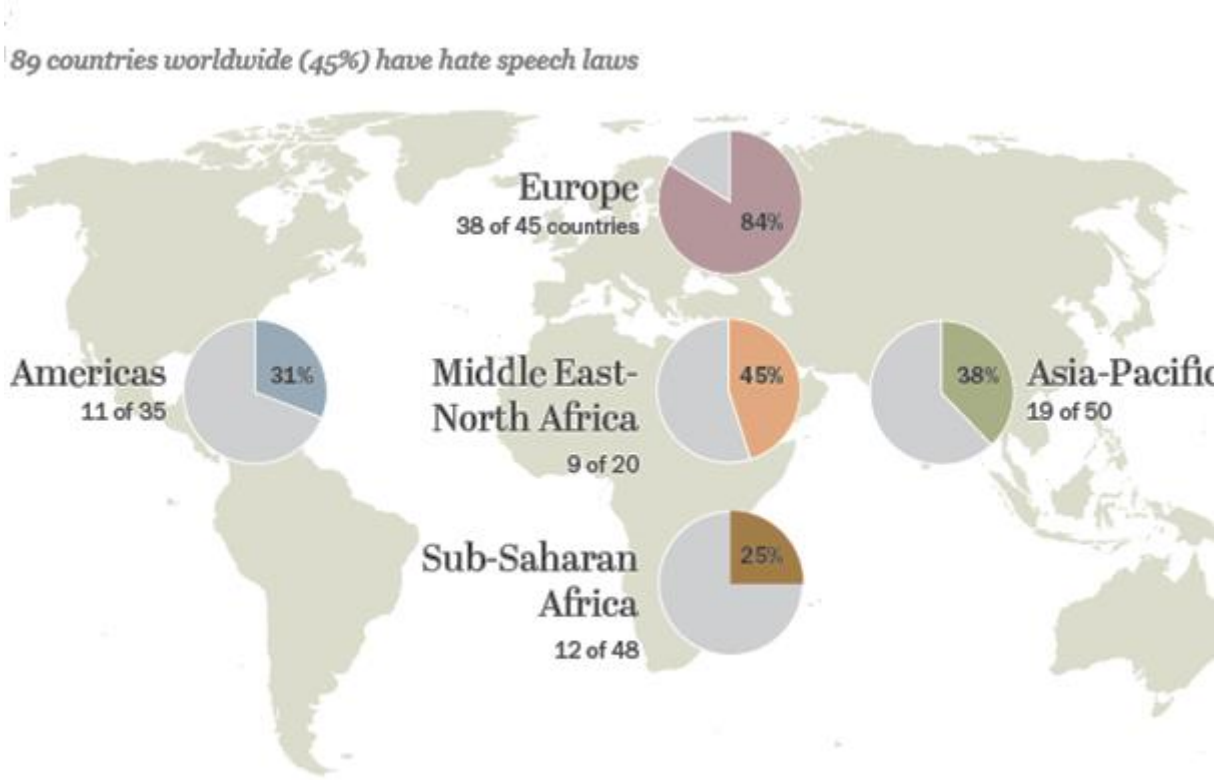


Figure 2: Hate speech laws around the world

4. Online hate speech and the Arabic language:

Recent years, the number of social media users is increasingly growing, especially in the Arabic region where users can freely express their opinion and thought which cause a high increasing rate of hate speech and hate crime then it becomes a challenge to detect and eliminate this offensive language from Arabic contents.

The Arabic language is the fourth most popular on the internet and the sixth most popular on Twitter. However, the Arabic language is a rich and complex language and it can be written in different forms and dialects. This is why it still has diverse challenges and open research topics.

4.1 Arabic in social media:

Most Arabic content on social media is written in an Arabic dialect, such as Syrian, Lebanese, Palestinian or Algerian, which is more difficult and ambiguous, that is why the research for Arabic dialects is still very limited due to the lack of resources needed.

4.2 Arabic dialect problems challenges:

- 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

(Alwakid et al., 2017)

5. Proposed solution:

5.1 The need of automatic hate speech detection:

Hate speech has become a more prevalent crime in recent years, as shown by increasing media coverage and political exposure to the issue. There are many causes to concentrate on automatic hate speech detection, which we explain in the followings:

- **European Union Commission directives:** The European Union Commission has undertaken a number of campaigns in recent years aimed at reducing hate speech. Some many programs are being founded to combat hate speech. Recently, the European Union Commission pressured Facebook, YouTube, Twitter, and Microsoft to sign an EU hate speech code, this requires the provision to check the majority of valid notifications in less than 24 hours for the elimination of illegal hate speech[10].
- **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(Ross et al., 2017). 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. However, detecting hate speech is a critical challenge since it is linked to real hate crimes (ILGA, 2016b), and automatic hate speech detection in text will also offer information about this phenomenon.
- **Removal of hateful speech:** Hate speech reporting and exclusion could be of concern to certain businesses and outlets. For example, online media publishers and online platforms in general need to draw advertisers and therefore cannot risk being labeled as hate speech platforms(Hewitt et al., 2016).Users may also be involved in blocking hate speech-related dialogue to prevent being vulnerable to it.

10 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.

- **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.

5.1.1 Automated Techniques for Hate Speech Detection:

During this work, we had the opportunity to discuss several projects about online hate speech detection. We present in the following the various recent works in this context divided into works which are interested in the study of English and Arabic languages. We notice that very few studies have been carried out for the Arabic dialects.

a. English datasets

(Williams & Burnap, 2016) used only 1901 tweets to train three binary classification algorithms: Bayesian Logistic Regression, Random Forest, and SVM for English hate speech. The tweets were annotated as offensive or not using crowdsourcing. A voting ensemble of the three classifiers, combined with a combination of word-ngram, reduced typed dependencies, generated the best overall result of 95%.

The study (Nobata et al., 2016b) used a binary regression model to evaluate a variety of features, including 1) ngram features, 2) linguistic features, 3) syntactic features, and 4) distributional semantics features like Word2vec. The researchers used two datasets of clean and offensive comments from Yahoo Finance and News to train a linear regression model. According to the reviewers, the model built on a mixture of syntactic and distributional semantics functions, as well as normal ngram features, outperformed single feature based models and obtained an overall F1-score of 82.6 percent.

Another research (Freund & Shah, 1994) used a Twitter dataset of 14,509 posts to train multiple classifiers to distinguish profanity from hate speech. The writers experimented with a variety of characters, word-ngrams, word skip-grams, and Brown clustering to extract features. They used three separate learning methods for the classification task: single-classifiers, ensemble classifiers, and meta-classifiers. However, using meta-classifiers, which merge several classifiers using a meta-learning technique, resulted in a classifier with the best accuracy of 79.8%.

An aggressive language recognition (OLID) dataset crawled from Twitter was recently proposed in research (Zampieri et al., 2019b). The dataset consists of 14,100 tweets that were manually labeled using crowdsourcing. Each tweet has three labels that indicate whether it is

offensive or not, and whether the target is a person or a group. On the three classification tasks, the authors compared the hybrid model of unigram+SVM to CNN and BILSTM neural networks. On all of the functions, the results obtained by the two neural network models were superior to those obtained by SVM, according to the experiment. However, CNN outperformed BILSTM in all three missions, with F1-scores of 80%, 69% and 47% for the three tasks respectively.

b. Arabic datasets

In 2017, (Mubarak et al., 2017) published one of the first research on Arabic abusive language identification, which included the development of an Arabic dataset of abusive comments. The dataset contains 1100 tweets that were gathered using divisive accounts and hashtags. Three annotators classified the dataset as offensive, abusive, and clean. The authors used pattern-based Twitter search to create a seed word list of 228 Arabic obscene terms for classification. Then, based on their use of offensive terms in the seed list, they divided Twitter users into clean and offensive categories. They also compiled a longer list of potentially offensive terms by removing only the unigram and bigram words used by the offensive community. They measured the utility of both lists as features for categorizing tweets as offensive or clean. The best F1-score of 60%.

In the paper (Albadi et al., 2018b), a more detailed dataset for religious hate speech in dialectical Arabic was presented. The dataset is made up of 6600 tweets gathered from Twitter using religious-related keywords. The authors looked at a number of features, including lexicon-based and n-gram features, as well as traditional machine learning algorithms. They have used neural networks like GRU and LSTM to test their theories. GRU had the highest estimation precision of 77%, according to the experiments.

The work (Mulki et al., 2019) created the Levantine hate speech (L-HSAB) Twitter dataset, which included 5812 tweets classified as common, hate, or violent. They compared SVM and Naive Bayes classification results and found that Naive Bayes outperformed SVM with an F1-score of 89.6% for 2-class classification and 74.4 percent for 3-class classification.

Participants competed for two binary Arabic text assignment subtasks: offensive (subtask A) and dislike (subtask B) in the OSACT4 shared task (Mubarak et al., 2020). (subtask B). The SemEval 2020 Mission 12 offensive language corpus, which consists of 10,000 Arabic tweets, was used for both tasks. LR, XGBoost, SVM, CNN, and RNN were used among a number of conventional and deep learning approaches. Furthermore, traditional embeddings such as fasttext and word2vec were included, as well as contextual embeddings such as BERT and Arabic Mazajak. Subtask A received the highest macro-averaged F1-score of 90.5%, while subtask B received 95.2 % from the same team.

They built a high-quality text corpus of Arabic hate and abusive expression in this (Alsafari et al., 2020). This is the first Arabic corpus of hate annotations. They performed a comprehensive

scientific study by testing a number of feature selection approaches, including machine and deep learning methods, within a supervised classification system. They used 2-class, 3-class, and 6-class datasets with heterogeneous feature spaces to train a large number of classifiers with an F1-macro of 85.16 percent, the mixture of word (1–3) and char (1–5) ngrams obtained the best precision in binary classification.

6. Challenges in Hate Speech Detection:

Hate speech is a complicated phenomenon that is difficult to identify. The authors of the surveyed articles identified the following obstacles and difficulties:

• Humans have a low level of consensus when classifying hate speech, implying that robots would have a tougher time. (Huang et al., 2018)

• Cultural and social structure knowledge is needed for this mission. (Raisi & Huang, 2016a)

• It's impossible to keep track of both ethnic and minority slurs when social phenomena and vocabulary evolve.(Nobata et al., 2016b)

• Language changes rapidly, particularly among young people who communicate frequently.

• Language evolves rapidly, particularly among young people who communicate frequently through social media. (Raisi & Huang, 2016b)

• Notwithstanding the fact that hate speech is unacceptable, inflammatory rhetoric can be very fluent and grammatically right, cross sentence boundaries, and contain sarcasm. (Nobata et al., 2016b)

• Finally, hate speech detection entails more than just looking for keywords. (Nobata et al., 2016b)

Conclusion:

In this chapter we discussed the hate speech detection problem, Firstly, we presented some definitions of hate speech and its different forms. Then we showed how abusive actions have a detrimental influence on users' physical and psychological wellbeing. In addition, we discussed about the current automated ways for dealing with toxicity in online spaces. Following that, we mentioned the difficulties that researchers faced in this subject.

CHAPTER 02:
Deep learning and transfer learning

Table of Contents

Chapter 02: **Deep learning and transfer learning** 15
Part1: 15
Introduction 15
1. Deep learning 15
1.1. Artificial Intelligence, Machine Learning and Deep Learning 15
1.2. Feature Learning 16
1.3 Types of deep learning approaches 16
1.3.1 Supervised learning 16
1.3.2. Unsupervised learning 16
1.3.3. Semi supervised learning 16
1.3.4. Deep Reinforcement Learning (RL) 17
1.4. Why and when to apply DL 17
1.5. Challenges of Deep learning 18
2. Deep neural network process 19
2.5. History 19
2.6. Gradient Descent 19
2.7. Stochastic Gradient Descent (SGD) 20
2.8. Back-Propagation (BP) 20
2.9. Activation fonction 20
2.9.1. Linear: 20
2.9.2. Sigmoid: 20
2.9.3. Tanh: 20
2.9.4. SoftMax: 20
2.9.5. ReLU: 21
2.10. Cost function 21
2.11. Regularization 22
2.11.1. Dropout 22
2.11.2. Drop connect 23
3. Convolutional Neural Network (CNN) 23
3.5. CNN Overview 23
3.6. Convolution Layer 23
3.7. Sub-sampling Layer 24
3.8. Classification Layer 24
4. Transfer learning 24

4.5. What Is a Pre-trained Model? 25
4.6. Why Will You Use Pre-Trained Models? 25
4.7. How Will You Use Pre-Trained Models? 25
5. Recurrent Neural Network (RNN) 26
6. Long Short-Term Memory (LSTM) 28
7. Gated Recurrent Unit (Gru) 29
8. Attention 30
8.5. Attention Mechanism 30
8.5.1. Self Attention 32
8.5.2. Global attention 32
8.5.3. Local attention 32
8.6. Transformer: 32
8.7. Auto encoder (AE) 34
9. Embedded Vectors: 36
9.5. Collobert and Weston (C&W) model: 36
9.6. Word2vec: 36
9.7. CBOW: 37
9.8. Skip gram: 37
9.9. FastText: 37
9.10. Global Vectors for Word Representation (GloVe): 38
9.11. Embeddings from Language Models (ELMo): 38
9.11.1. ELMo Architecture: 39
9.12. Graphs from LOw-level unit Modeling (GLoMo): 39
9.13. Universal Language Model Fine-tuning (ULMFiT): 40
9.14. Bidirectional BERTEncoder Representation from Transformers (BERT): 40
9.14.1. Model Architecture: 40
9.14.2. BERT Input Representation: 41
9.14.3. BERT Output Representation: 42
Part2: 43
1. Related work: 43
1.1. Current state in Hate Speech Detection and Related Concepts: 43
1.2. Abusive language detection: 43
1.3. Cyber bullying detection: 43
1.4. Hate speech detection: 44
1.5. Arabic Hate Speech Detection: 44

1.5.1. Arabic anti-social behavior detection: 45
1.5.2. Arabic hate speech detection: 46
1.5.3. Algerian dialect hate speech detection: 46
1.6. Summary and Analysis 47
Conclusion 56

Part1:

Introduction

Deep learning has seen a lot of success in the last several years across a wide range of applications. Machine learning is a fast-expanding discipline that has been used to a wide range of traditional applications. as well as some new areas that present more opportunities. The development of a system to detect abusive language online is critical for Internet users' health and safety. Cyberbullying, online harassment, and other technological abuses are on the rise, according to studies. In 2020, particularly during the worldwide Coronavirus epidemic, 35% of respondents reported online abuse based on their identity-based features, up 3% from 2019.

Applying advanced techniques from the Natural Language Processing (NLP) field to support the development of an online hate free community is a critical task for social justice. Transfer learning enhances the performance of the classifier by allowing the transfer of knowledge from one domain or one dataset to others that have not been seen before, thus, supporting the classifier to be more generalizable.

This chapter focuses on the proposed deep learning-based approaches for dealing with hate speech problems that have been published in the literature. We explain what deep learning is and how it differs from machine learning in the first section of this chapter, as well as some new deep learning trends such as transfer learning and ensemble learning.

Then we go through deep learning architectures and current embeddings for textual data representation, and finally we go over novel architectures for language understanding that have recently arisen in the NLP field. In the second section, we discuss some relevant work in both English and Arabic, as well as Algerian dialects.

1. Deep learning

1.1. Artificial Intelligence, Machine Learning and Deep Learning

Artificial intelligence (AI) is a branch of technology in which researchers and scientists create machines that mimic human intelligence and behaviors. Machine learning (ML) is a branch of AI concerned with enabling computers to learn without being directly programmed. Deep learning (Dl) is a subset of machine learning approaches that are built on artificial neuron networks that are capable of adapting to new data.

1.2. Feature Learning

A key difference between traditional ML and DL is in how features are extracted. Traditional ML approaches use handcrafted engineering features by applying several feature extraction algorithms, and then apply the learning algorithms. Additionally, other boosting approaches are often used where several learning algorithms are applied to the features of a single task or dataset and a decision is made according to the multiple outcomes from the different algorithms. On the other hand, in the case of DL, the features are learned automatically and are represented hierarchically in multiple levels. This is the strong point of DL against traditional machine learning approaches. (Bengio, 2009)

1.3 Types of deep learning approaches

The following are the several types of deep learning approaches: Unsupervised, semi-supervised, or partially supervised. In addition, there is a type of learning approach known as Reinforcement Learning (RL) or Deep Reinforcement Learning (DRL), which is commonly considered as part of semi-supervised or unsupervised learning techniques. (Alpaydin, 2020; Schmidhuber, 2015)

1.3.1 Supervised learning

Supervised learning is a strategy for learning that employs labeled data. The environment contains a collection of inputs and outputs in supervised deep learning algorithms. Deep Neural Networks (DNN), Convolution Neural Networks (CNN), Recurrent Neural Networks (RNN), including Long Short Term Memory (LSTM), and Gated Recurrent Units are some of the many supervised learning methodologies for deep learning (GRU). (Alpaydin, 2020; Schmidhuber, 2015)

1.3.2. Unsupervised learning

Data labels aren't required for unsupervised learning systems to perform. Often clustering, dimensionality reduction, and generative techniques are considered as unsupervised learning approaches. Auto-encoders (AE), and the recently discovered GAN are all members of the deep learning family that are effective at clustering and non-linear dimensionality reduction. RNNs, such as LSTM and RL, are also utilized for unsupervised learning in a wide range of applications. (Alpaydin, 2020; Schmidhuber, 2015)

1.3.3. Semi supervised learning

Learning that is based on partially labeled datasets is known as semi-supervised learning. DRL and Generative Adversarial Networks (GAN) are semi-supervised learning approaches that are employed in some instances. RNNs, which include LSTM and GRU, are also used for semi-supervised learning. (Alpaydin, 2020; Schmidhuber, 2015)

1.3.4. Deep Reinforcement Learning (RL)

Deep Reinforcement Learning is a strategy for learning in unknown environments. DRL began in 2013 with Google DeepMind (Ackley et al., 1985; Bengio et al., 2013). Since then, various advanced methods have been presented based on RL. This method is sometimes referred to as semi-supervised learning. In RL, we do not have a straight forward loss function, thus making learning harder compared to traditional supervised approaches. The primary distinctions between supervised and RL are: You don't have complete access to the function you're trying to optimize; you have to query it through interaction, and you're also engaging with a state-based environment: Previous actions have an impact on input xt. (Alpaydin, 2020; Schmidhuber, 2015)

1.4. Why and when to apply DL

DL is used in a variety of instances where machine intelligence is beneficial:

• Absence of a human expert (navigation on Mars).

•Humans are unable to articulate their knowledge (speech recognition, vision, and language understanding).

• The problem's answer evolves with time (tracking, weather prediction, preference, stock, price prediction).

• Adapting solutions to specific situations is required (biometrics, personalization).

•The problem is just too large for our limited thinking powers to handle (calculation webpage ranks, matching ads to Facebook, sentiment analysis).

1.5. Challenges of Deep learning

There are several challenges for DL:

• Big data analytics using DL

• The ability of DL approaches to scale

•Ability to produce data, which is critical when data is unavailable for system learning. (Especially for computer vision tasks, such as inverse graphics).

• Energy efficient techniques for special purpose devices, including mobile intelligence, and so on.

• Multi-task and transfer learning or multi-module learning. This means learning from different domains or with different models together.

• Dealing with causality in learning

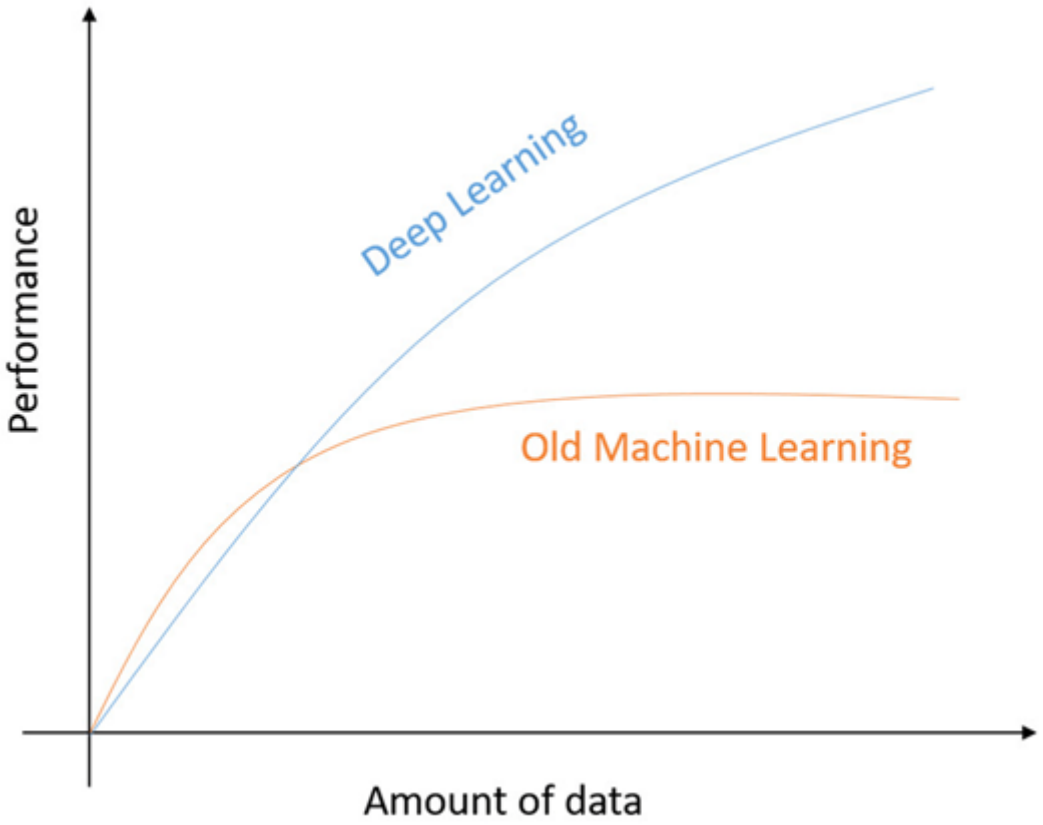


Figure 3: The performance of deep learning with respect to the amount of data(Alom et al., 2019)

2. Deep neural network process

2.5.History

A brief history of neural networks highlighting key events is:

•1943: (McCulloch & Pitts, 1943) show that neurons can be combined to construct a Turing machine

•1957: (Rosenblatt, 1958) shows that perceptron will converge if what they are trying to learn can be represented.

•1969: (Minsky & Papert, 1988) show the limitations of perceptron, killing research in neural networks for a decade.

•1985: The backpropagation algorithm by (Ackley et al., 1985). revitalizes the field.

•1988: (Fukushima, 1988) a hierarchical neural network capable of visual pattern recognition.
•1998: CNNs with Backpropagation for document analysis by (Lecun et al., 1998) .

•2006: The Hinton lab solves the training problem for DNNs. (G E Hinton & Salakhutdinov, 2006; Geoffrey E Hinton et al., 2006)

•2012: present a variety of deep learning algorithms are increasingly emerging.

Computational neurobiology has done a lot of work on creating computer models of artificial neurons. Artificial neurons, which attempt to replicate human brain functioning, are a key component in the development of ANNs. The basic computational element (neuron) is called a node (or unit) which receives inputs from external sources and has some internal parameters. This unit is called a perceptron. (Bengio et al., 2013; Schmidhuber, 2015)

ANNs or general NNs consist of Multilayer Perceptron's (MLP) which contain one or more hidden layers with multiple hidden units (neurons) in them. (Ackley et al., 1985; Bengio et al., 2013; Schmidhuber, 2015)

2.6. Gradient Descent

The gradient descent has been effectively utilized to train ANNs in the previous couple of decades. This method is a first-order optimization strategy for locating the local minima of a function. (Bengio et al., 2013; Schmidhuber, 2015)

2.7. Stochastic Gradient Descent (SGD)

Because the standard gradient descent methodology has a long training period, the SGD methodology is utilized to train Deep Neural Networks (DNN). (Bottou, 2012; Schmidhuber, 2015)

2.8.Back-Propagation (BP)

With SGD, DNN is trained using the popular Back-Propagation (BP) algorithm (Ackley et al., 1985; Kober et al., 2013). We can simply express NN models using computation graphs, which are directed acyclic graphs, in the case of MLPs. For that representation of DL, we can use the chain-rule to efficiently calculate the gradient from the top to the bottom layers with BP. (Ackley et al., 1985; Mohamed et al., 2012; Rumelhart et al., 1986)

2.9.Activation fonction

2.9.1. Linear:

The linear activation function is essentially the identity function, with inputs multiplied by weights for each neuron to produce an output. Backpropagation is not possible with this type of activation. (Torres et al., 2018)

2.9.2. Sigmoid:

It's a nonlinear activation that allows legitimate data to be reduced in extreme or unusual values without being eliminated: it turns independent variables with nearly limitless ranges into simple probabilities between 0 and 1. The majority of its output will be extremely close to 0 or 1. (Torres et al., 2018)

2.9.3. Tanh:

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. (Torres et al., 2018)

2.9.4. SoftMax:

It gives the probability distribution across mutually exclusive output classes and is frequently seen in the output layer of a neural network. (Torres et al., 2018)

2.9.5. ReLU:

The activation function rectified linear unit (ReLU) is a fascinating transformation that turns on a single node when the input exceeds a specified threshold. The more common and default 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. (Torres et al., 2018)

2.10. Cost function

1) Cross-entropy Loss:

The performance of a classification model whose output is a probability value between 0 and 1 is measured by logistic loss. As the projected likelihood differs from the actual label, cross-entropy loss grows. The log loss in an ideal model would be 0.3. It may be expressed mathematically as demonstrated in formula (1).

$$\mathrm{CE} = -\sum \mathrm{ti}\ (\mathrm{si})\ \mathrm{C}\ \mathrm{I} \qquad (1)$$

Where $\boldsymbol{ti}$ and $\boldsymbol{si}$ are respectively the ground-truth and the model score of a sample for the class i. (Vaswani et al., 2017)

2) Categorical Cross-entropy Loss:

Also called SoftMax loss. It is SoftMax activation plus cross-entropy loss and it is used for multi-class classification where the labels are one-hot. Its formula is written like (2):

$$\mathrm{CCE} = -(\ \mathrm{e}\ \mathrm{sp} \sum \mathrm{e}\ \mathrm{s}\ \mathrm{C}\ \mathrm{j}\ \mathrm{j}) \qquad (2)$$

Where $\boldsymbol{sp}$ is the model score for the positive class of the given sample. (Vaswani et al., 2017)

3) Focal Loss:

was introduced by (Tsung-Yi Lin, 2017) from Facebook. The one-stage object detection situation, in which the foreground and background classes are extremely imbalanced during training in dense object identification. It is a cross-entropy loss that weighs the contribution of each sample to the loss based on the classification error. In addition, the authors also weigh the contribution of each class to the loss in a more explicit class balancing. They added to cross entropy $(\mathbf{1} - \boldsymbol{Pt})\,\boldsymbol{\gamma}$ as a modulating factor to reduce the influence of correctly classified samples in the loss and with tunable focusing parameter $\boldsymbol{\gamma} \geq \mathbf{0}$ they defined it as:

$$\mathrm{FL} = -\alpha t\,(1 - \mathrm{Pt})\,\gamma \log(\mathrm{Pt}) \qquad (\mathbf{3})$$

Where $\boldsymbol{y}$ is the true label of the sample, $\boldsymbol{p}$ is the model prediction; the factor $(\boldsymbol{Pt})$ is the rewritten CE by defining $\boldsymbol{Pt}$ as in the formula (4) below:

$$\mathrm{Pt} = \{, \mathrm{y} = 1\ 1 - \mathrm{p}, \mathrm{otherwise} \qquad (4)$$

$\boldsymbol{\alpha t}$ is weighting factor used for addressing class imbalance with $\boldsymbol{\alpha} \in [\mathbf{0}, \mathbf{1}]$ and:

$$\alpha t = \{, \mathrm{y} = 1\ 1 - \alpha, \mathrm{otherwise} \qquad (5)$$

2.11. Regularization

2.11.1. Dropout

Different regularization algorithms for deep CNN have been presented in the last several years. (Geoffrey E. Hinton et al., 2012) suggested the simplest but most efficient technique termed dropout. A randomly selected subset of activations inside a layer is set to zero in Dropout. Figure 4 shows the dropout notion.

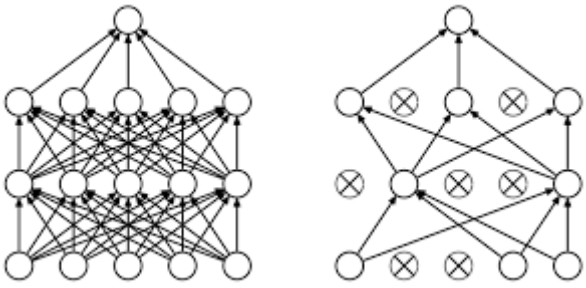


Figure 4: Pictorial representation of the concept Dropout.

2.11.2. Drop connect

Drop Connect is another regularization technique. Instead of removing the activation, the subset of weights within the network layers is set to zero in this scenario. As a consequence, each layer receives a subset of units from the preceding layer that was chosen at random (Wan et al., 2013).

3. Convolutional Neural Network (CNN)

3.5.CNN Overview

(Fukushima, 1988) presented this network layout in 1988 for the first time. However, because of computational hardware limitations, it was not frequently employed for network training. In the 1990s, (Lecun et al., 1998) applied a gradient-based learning algorithm to CNNs and obtained successful results for the handwritten digit classification problem. After that, researchers further improved CNNs and reported state-of-the-art results in many recognition tasks. CNNs offer various benefits over DNNs, including being more like the human visual processing system, having a highly optimized structure for processing 2D and 3D pictures, and being more like the human visual processing system and being effective at learning and extracting abstractions of 2D features. CNNs' max pooling layer is good at absorbing shape fluctuations. CNNs also contain fewer parameters than a fully connected network of the same size since they are made up of sparse connections with coupled weights. Most importantly, CNNs are learned using a gradient-based learning method, which means they are less susceptible to the decreasing gradient issue. CNNs may yield highly optimized weights since the gradient-based technique directly trains the whole network to minimize an error criterion.

3.6.Convolution Layer

Feature maps from preceding layers are convolved using learnable kernels in this layer. The output feature maps are formed by passing the kernel output via a linear or non-linear activation function (such as sigmoid, hyperbolic tangent, SoftMax, rectified linear, and identity functions). More than one input feature map can be mixed with each of the output feature maps. In general, we have that

$$x_j^l = f\left(\sum_{iMj} x_i^{l-1} * k_{ij}^l + b_j^l\right) \quad (6)$$

where *xl* is the output of the current layer, x^{l-1} is the previous layer output, *kl is* the kernel for the present layer, and *bl* are the biases for the current layer. M_j represents a selection of input maps. For each output map, an additive bias *b* is given. However, the input maps will be convolved with distinct kernels to generate the corresponding output maps. The output maps finally go through a linear or non-linear activation function (such as sigmoid, hyperbolic tangent, SoftMax, rectified linear, or identity functions).

3.7.Sub-sampling Layer

The down-sampled process is performed on the input maps by the subsampling layer. This is commonly known as the pooling layer. The number of input and output feature maps does not change in this layer. There will be precisely N output maps if there are N input maps. As a result of the down sampling procedure, according to the size of the down sampling mask, the size of each dimension of the output maps will be decreased. If a 2* 2 down sampling kernel is employed, for example, each output dimension for all the pictures will be half of the corresponding input dimension. This operation can be formulated as

$$x_j^l = \text{down}\,(x_j^{l-1}) \tag{7}$$

where down(.) represents a sub-sampling function. This layer primarily performs two sorts of operations: average pooling and max pooling. In the average pooling strategy, the function typically adds across N* N patches of the preceding layer's feature maps and chooses the average value. In the case of max-pooling, however, the greatest value is chosen from the N N patches of the feature maps. As a result, the output map dimensions are n times smaller. Each output map is multiplied with a scalar in some exceptional instances. Alternative sub-sampling layers, such as fractional max-pooling and subsampling using convolution, have been proposed.

3.8.Classification Layer

This is the fully connected layer that calculates each class's score using the retrieved features from the convolutional layer in the previous phases. The last layer feature maps are supplied to the fully connected layers as vectors with scalar values. As a soft-max classification layer, fully connected feed-forward neural layers are utilized. There are no hard and fast rules on how many layers should be included in a network model. However, in most cases, two to four layers have been observed in different architectures, including LeNet(Lecun et al., 1998), AlexNet (Geoffrey E. Hinton et al., 2012), and VGG Net (Simonyan & Zisserman, 2015). As the fully connected layers are expensive in terms of computation, alternative approaches have been proposed during the last few years. These include the global average pooling layer and the average pooling layer which help to reduce the number of parameters in the network significantly. The completely connected layer updates in the backward propagation across the CNNs, following the usual methodology of fully connected neural networks (FCNN). By executing the whole convolutional operation on the feature maps between the convolutional layer and its immediate prior layer, the filters of the convolutional layers are changed.

4. Transfer learning

Examining the student-teacher interaction is an excellent technique to understand transfer learning. After gaining extensive information about a subject, a teacher gives a course (Pan & Yang, 2010). The content will be delivered over time through a series of lectures. The instructor

(expert) is imparting information (knowledge) to the students in this way (learner). In deep learning, the same process happens: a network is trained with a large quantity of data, and the model learns the weights and biases while training. These weights may be used to test or retrain a comparable new model in other networks. The network can start with pre-trained weights instead of training from scratch. The conceptual diagram for transfer learning method is shown in Figure 5.

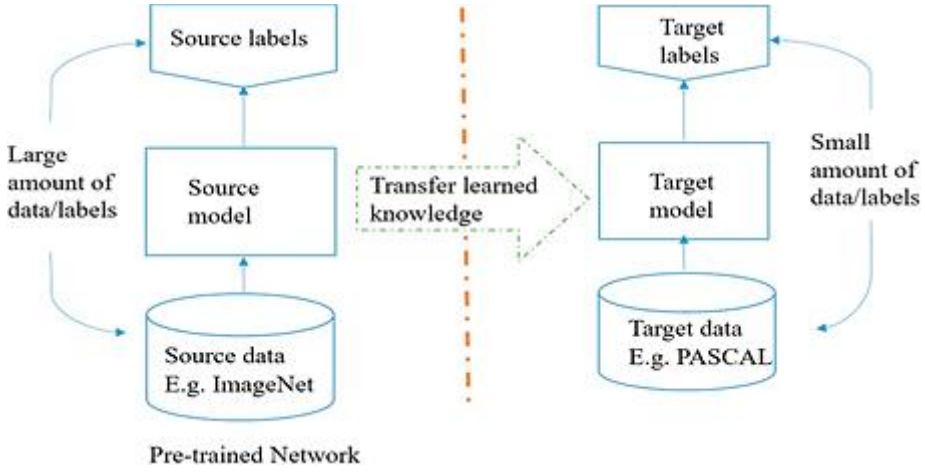


Figure 5: Conceptual diagram for transfer learning: Pretrained on ImageNet and transfer learning is used for retraining on PASCAL dataset.

4.5.What Is a Pre-trained Model?

A pre-trained model is one that has already been trained in the same domains as the one for which it is being used. For example, an Inception model that has previously been trained on ImageNet may be downloaded for an image identification challenge. Instead of training the Inception model from scratch, the weights can be kept as is with some acquired features, and the model may be utilized for a new recognition problem. When there is a lack of sample data, this approach of training is effective.

4.6.Why Will You Use Pre-Trained Models?

There are several advantages to adopting pre-trained models. To start with, training large models on large datasets necessitates a lot of expensive processing resources. Second, training large models can take up to several weeks. Convergence and network generalization can be aided by training new models using pre-trained weights.

4.7.How Will You Use Pre-Trained Models?

We need to consider the following criterions with respective application domains and size of the dataset when using the pre-trained weights which are shown in Table 2.

Table 2 Criterions need to be considered for transfer learning.

Methods	New Dataset but Small	New Dataset but Large
Pre-trained model on similar but new dataset	Freeze weights and train linear classifier from top level features	Fine-tune all the layers (pre-train for faster convergence and better generalization)
Pre-trained model on different but new dataset	Freeze weights and train linear classifier from non-top-level features	Fine-tune all the layers (pre-train for enhanced convergence speed)

5. Recurrent Neural Network (RNN)

Human can understand each word or sentence based on the understanding of previous words or sentences. Traditional neural network techniques, such as DNNs and CNNs, are unable to solve this situation. Because of the following reasons, conventional Neural Networks and CNN are ineffective. To begin with, these approaches only accept a fixed-size vector as input (e.g., an image or video frame) and output a fixed-size vector (e.g., probabilities of different classes). Second, those models operate with a fixed number of computational steps (e.g., the number of layers in the model). The RNNs are unique as they allow operation over a sequence of vectors over time. (Sathasivam & Wan Abdullah, 2008) introduced this concept in 1982 but the idea was described shortly in 1974. The pictorial representation is shown in Figure 6.

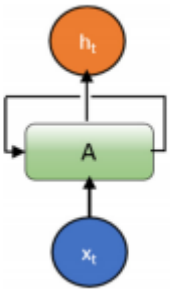


Figure 6: The structure of basic Recurrent Neural Network (RNN) with a loop.

(Elman, 1990; Jordan, 1997) presented several different RNN variants. The Elman design employs hidden layer outputs as inputs in addition to hidden layer regular inputs (Tsung-Yi Lin, 2017). In Jordan's network (X. Wang et al., 2017), on the other hand, the output unit's outputs are utilized as inputs with the hidden layer's inputs. Mathematically expressed as:

(Elman, 1990):

$$h_t = \sigma_h(w_h x_t + u_h h_{t-1} + b_h) \quad (8)$$

$$y_t = \sigma_y(w_y h_t + b_y) \quad (9)$$

(Elman, 1990; Jordan, 1997):

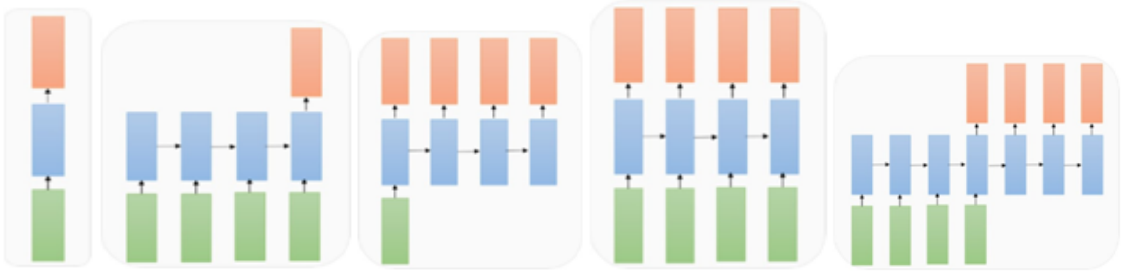
$$h_t = \sigma_h(w_h x_t + u_h y_{t-1} + b_h) \quad (10)$$

$$y_t = \sigma_y(w_y h_t + b_y) \quad (11)$$

where xt is a vector of inputs, ht are hidden layer vectors, yt are the output vectors, w and u are weight matrices and b is the bias vector.

From an application standpoint, RNNs may tackle a variety of issues that need diverse RNN topologies, as shown in Figure 7. In Figure 7, input vectors are shown in green, RNN states are shown in blue, and the output vector is shown in orange. These structures can be summarized as follows:

- **One to One**: Standard mode for classification without RNN (e.g., image classification problem) shown Figure 7. (a)
- **Many to One:** Sequence of inputs and a single output (e.g., the sentiment analysis where inputs are a set of sentences or words and output is a positive or negative expression) shown Figure 7. (b).
- **One to Many:** Where a system takes an input and produces a sequence of outputs (Image Captioning problem: Input is a single image and output is a set of words with context) shown Figure 7. (c).
- **Many to Many**: Sequences of inputs and outputs (e.g., machine translation: machine takes a sequence of words from English and translates to a sequence of words in French) shown Figure 7. (d).
- **Many to Many:** Sequence to sequence learning (e.g., video classification problem in which we take video frames as input and wish to label each frame of the video shown Figure 7. (e).



(a) (b) (c) (d) (e)

Figure 7: The different structure of RNN with respect to the applications: (a) One to one; (b) many to one; (c) one to many; (d) many to many; and (e) many to many

Information can be transmitted from one network step to the next via a loop. A recurrent neural network is made up of several copies of the same network, each transmitting a message to the next. Figure 8, illustrates what occurs when we unroll the loop.

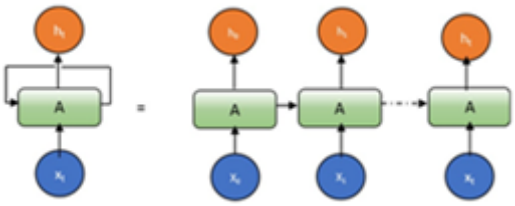


Figure 8: Unrolled RNNs

The vanishing gradient problem is the most significant drawback of RNN techniques. (Hochreiter & Frasconi, 2001) are the first to overcome this challenge. In 1993, a deep RNN with 1000 successive layers was constructed and assessed to tackle deep learning problems. For tackling the vanishing gradient problem of RNN techniques, several approaches have been proposed. The first is to clip and scale the gradient if the norm is too large, and secondly, create a better RNN model. One of the better models was introduced by (F A Gers & Schmidhuber, 2000; Felix A. Gers et al., 2003) named Long Short-Term Memory (LSTM) From the LSTM there have been different advanced approaches proposed in the last few years.

The RNN approaches allow sequences in the input, the output, or in the most general case both. For example, DL for text mining, building deep learning models on textual data requires representation of the basic text unit and word. Structures of neural networks that may hierarchically represent the text's sequential nature. RNNs, or Recursive Neural Networks, are used for language understanding (Socher et al., 2011). In language modeling, it attempts to predict the next word or collection of words, or even sentences, based on the preceding ones (Ingolfsson & Sachs, 1993). RNNs are networks that contain loops, which allow information to endure. Another example: RNNs may relate earlier knowledge to the current job by analyzing prior video frames, comprehending the present, and attempting to create future frames (Chao et al., 2018).

6. Long Short-Term Memory (LSTM)

The key idea of LSTMs is the cell state, the horizontal line running through the top of Figure 9. LSTMs remove or add information to the cell state called gates: An input gate (it), forget gate (ft) and output gate (ot) can be defined as (formula 12,13,14,15,16,17):

$$f_t = \sigma(w_f . [h_{t-1}, x_t] + b_f) \quad (12)$$

$$i_t = \sigma(w_i . [h_{t-1}, x_t] + b_i) \quad (13)$$

$$\tilde{c}_t = \tanh(w_c . [h_{c-1}, x_t] + b_c) \quad (14)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (15)$$

$$o_t = \sigma(w_o.[h_{t-1}, x_t] + b_o) \quad (16)$$

$$h_t = o_t * \tanh(c_t) \quad (17)$$

LSTM models are popular for temporal information processing. Most of the papers include LSTM models with some minor variance. There is a slightly modified version of the network with peephole connections by (F A Gers & Schmidhuber, 2000) proposed in 2000. The concept of peepholes is included with almost all the gates in this model.

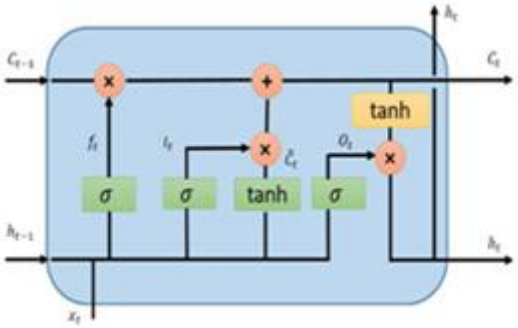


Figure 9: Diagram for Long Short-Term Memory (LSTM).

7. Gated Recurrent Unit (Gru)

GRU was similarly derived from LSTMs, but with a little more variety (Chung et al., 2014). Those who work with recurrent networks are increasingly turning to GRUs. The model's popularity stems mostly from its low computational cost and simplicity, as seen in Figure 10. GRUs are lighter versions of RNN approaches than standard LSTM in term of topology, computation cost and complexity (Chung et al., 2014). This technique combines the forget and input gates into a single update gate and merges the cell state and hidden state along with some other changes. The simpler model of the GRU has been growing increasingly popular. Mathematically the GRU can be expressed with the following equations:

$$z_t = \sigma(w_z.[h_{t-1}, x_t]) \quad (18)$$

$$r_t = \sigma(w_r.[h_{t-1}, x_t]) \quad (19)$$

$$\check{h}_t = \tanh(w.[r_t * h_{t-1}, x_t]) \quad (20)$$

$$\mathrm{h_t} = (1 - \mathrm{z_t}) * \mathrm{h_{t-1}} + \mathrm{z_t} * \tilde{\mathrm{h}}_\mathrm{t} \quad (21)$$

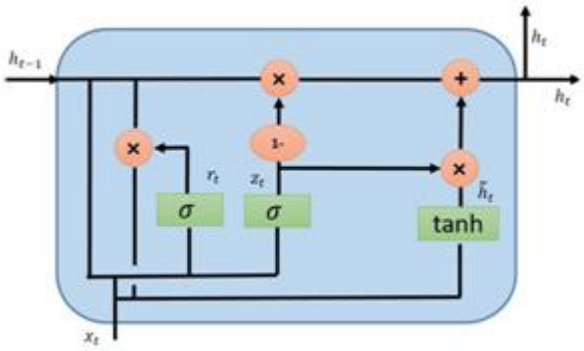


Figure 10 : Diagram for Gated Recurrent Unit (GRU)

8. Attention

(Bahdanau et al., 2015) was the first to present the Attention Mechanism for machine translation, and it was further developed by (Vaswani et al., 2017). Its goal is to assess the relevance of each hidden state and provide a weighted total of all the input characteristics. This allows typical restrictions to be solved in a variety of NLP applications. As an example, traditional word vectors presume that a word's meaning is relatively stable across sentences. However, A single word can have many meanings depending on the context, such as lit (a blazing adjective) and lit (a literary abbreviation); or get (a verb for getting) and get (an animal's progeny). ALs can capture the several shades of meaning that a word might have depending on where it appears in a sentence and how it interacts with other words. Furthermore, ALs learn how to link an input sequence to the model's output in order to focus on more relevant input features. As a result of this technique, models are able to mimic the human brain's ability to focus on certain sections of the input, allowing them to determine the relevance of each portion for the intended output. (Dessí et al., 2020) For more details of how this attention layer works, we take as example the attention layer used in (Bahdanau et al., 2015) work and that used in the Transformer architecture of (Vaswani et al., 2017) for machine translation.

8.5.Attention Mechanism

For the first time, attention was added as an addition to a Sequence-to-Sequence architecture for machine translation (Bahdanau et al., 2015). In this contribution, it is proposed that attention to jointly align and translate, where alignment refers to the problem in machine translation of determining which parts of the input sequence are relevant to each word in the output, Translation, on the other hand, is the process of selecting the right output based on the required information.

Previous work (Cho et al., 2014) addressed this issue by proposing encoder-decoder architecture as machine translation and incorporating GRUs. In this project, it was recommended that the input sequence be encoded into a single fixed context vector that could be utilized by the decoder to generate the output sequence, which would represent the input's translation. In reality, representing the whole input sequence in a single vector does not give the decoder enough information about the important sections of the input to conduct appropriate

translations as a result, (Bahdanau et al., 2015) considered putting a single hidden layer neural network between the encoder and the decoder to simulate a context vector that is filtered particularly for each output time step and provides information about which portions of the input sequence are significant.

A single attention layer is inserted between the encoder Bi-RNN and decoder RNN layers in a similar scheme. The architecture utilized in Figure 11 shows how all feature vectors hj from the encoder Bi-RNN are supplied to attention. The context vector ct produced by the attention layer is passed to the decoder RNN layer. The context vector is a weighted sum of the encoder layer's hidden states, with attention weights at,j (formula 22).

$$ct = \sum at,* \ hj \ T \ j = 1 \tag{22}$$

Contribution of the feature vector is computed using softmax where the attention weights $\boldsymbol{at}$, sum to 1, and all features contribute to decoder at each time step t. Formula 23and 24 below show how the attention contribution is calculated

$$at, = \ e(et, j) \sum \exp(et, k) \ T \ k = 1 \tag{23}$$

Where:

$$et, j \ = \ a(st - 1, hj) \tag{24}$$

et, is an alignment model which scores how well the inputs around position j and the output at position t match. The score is based on the RNN hidden state $st-1$. The words yt are generated by using a conditional probability that rely on a different context vector in each time step (Bahdanau et al., 2015).

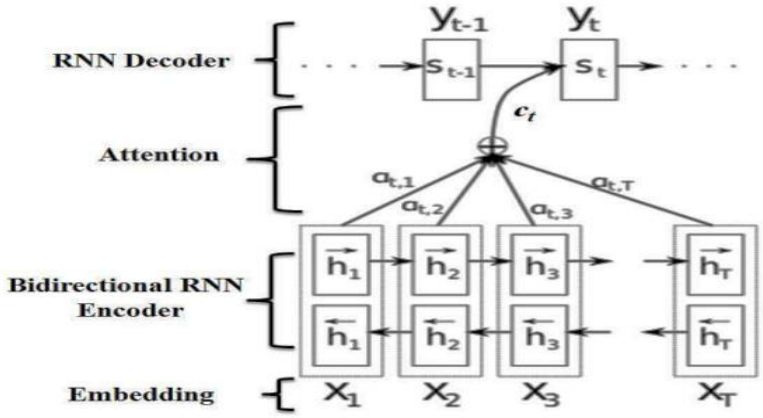


Figure 11: Sequence to Sequence Model architecture using Attention Mechanism. (Bahdanau et al., 2015)

8.5.1. Self Attention

Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. It has been shown to be very useful in machine reading, abstractive summarization, or image description generation. .(Luong et al., 2015)

8.5.2. Global attention

The idea of a global attentional model is to consider all the hidden states of the encoder when deriving the context vector. Figures 12 show the global attention. .(Luong et al., 2015)

8.5.3. Local attention

The global attention has a drawback that it has to attend to all words on the source side for each target word, which is expensive and can potentially render it impractical to translate longer sequences, e.g., paragraphs or documents. To address this problem, local attentional mechanism chooses to focus only on a small subset of the source positions per target word. Figures 12 show the local attention. (Luong et al., 2015)

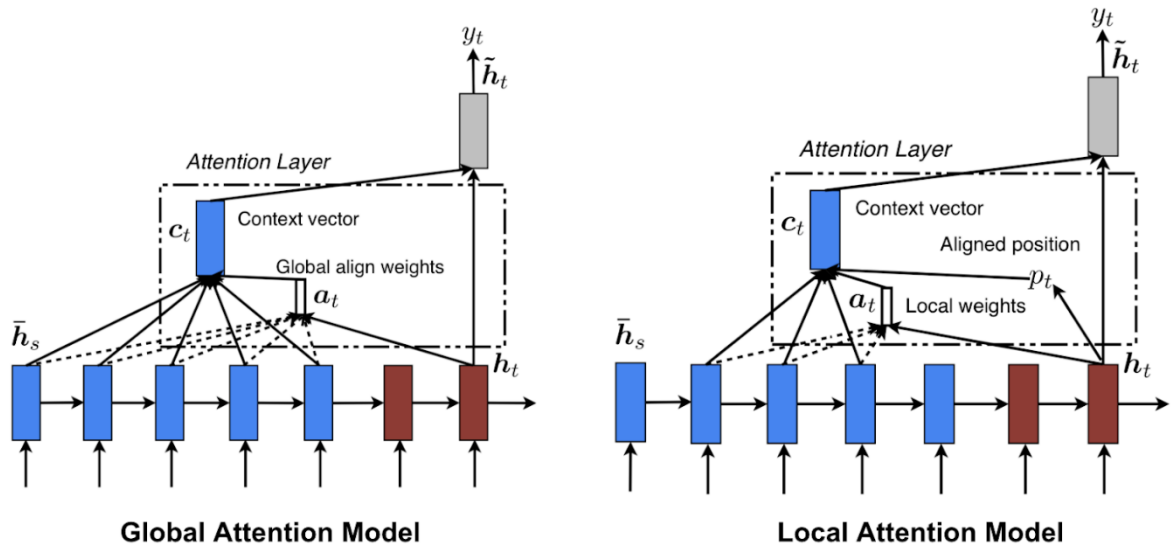


Figure 12: Global and local attention model*(Luong et al., 2015)*

8.6.Transformer:

A transformer is an encoder-decoder architecture topped by linear (totally connected neural network) and SoftMax layers, with each portion (encoder and decoder) made up of a stack of six identical layers. One layer of this stack is made up of a multi-head attention layer atop a feed-forward network for the encoder component, with an extra layer that performs multi-head attention over the output in the decoder. (Vaswani et al., 2017).

Each place in the encoder can attend to all positions in the encoder's preceding layer.

Similarly, decoder self-attention layers allow each location in the decoder to attend to all positions in the decoder up to and including that position. The decoder's leftward information flow is controlled in order to retain the auto-regressive function. (Vaswani et al., 2017).

This indicates that the encoder has bidirectional conditioning in both directions, but the decoder only has unidirectional conditioning in one direction, from left to right.

The transformer architecture employs 8-head attention layers, with each head supplied by embeddings of sentence tokens with a positional encoding that identifies the token's place in the phrase. Then, for each token embedding, three vectors Q, K, and V are produced by multiplying the token embedding by three weights matrices WQ, WK, and WV that are trained throughout the training phase.

The attention-head layer generates a vector representation for each token embedding based on the other relevant tokens in the sentence by multiplying the VQuery of one token by the VKey of the other tokens, then calculating the SoftMax for the resulting value (Q*Kt) divided by the square root of the dimension of the key vectors. This SoftMax reflects the degree to which each sentence token is relevant to the provided term.

Finally, the total of the value vectors (V) multiplied by the related SoftMax value is used to generate the head-attention output.

Encoder: 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)

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 13 show the encoder and decoder architecture.

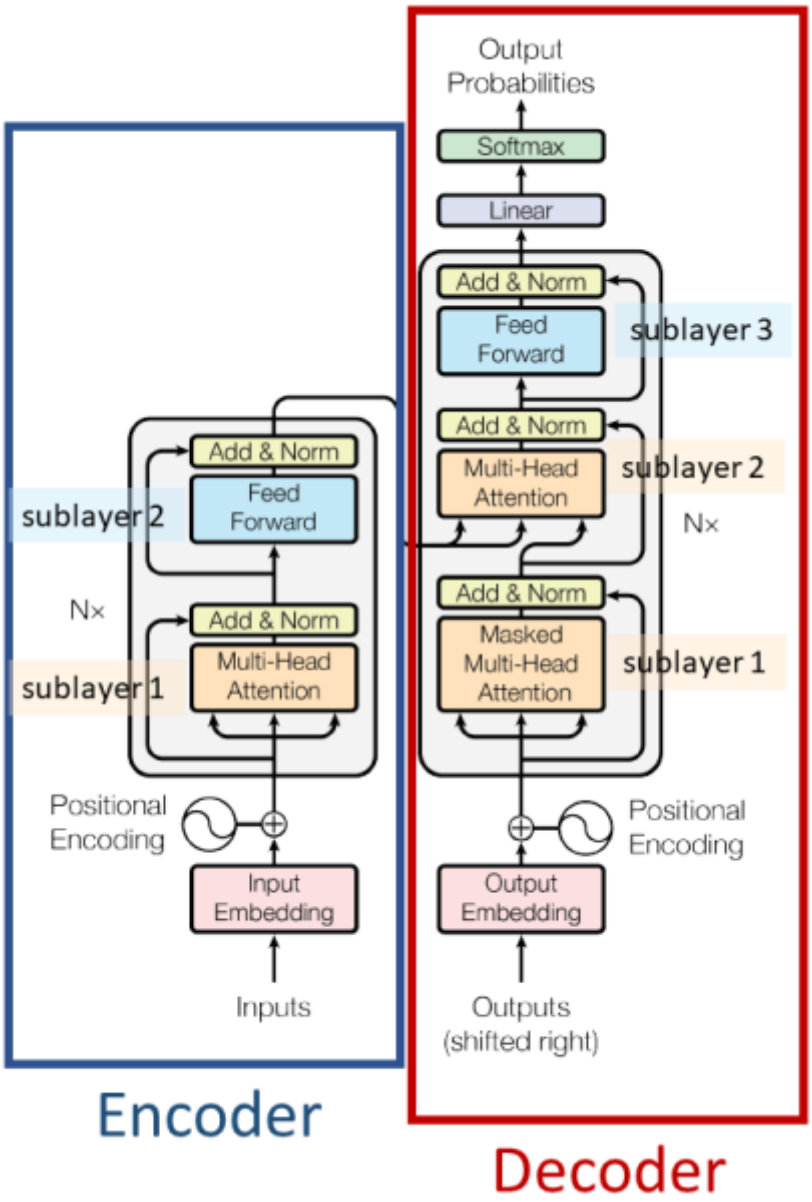


Figure 13: The Transformer model architecture. (Vaswani et al., 2017)

8.7.Auto encoder (AE)

A deep neural network strategy for unsupervised feature learning with efficient data encoding and decoding is known as an AE. The primary goal of an auto encoder is to learn and represent (encode) the input data, which is often used for data dimensionality reduction, compression, fusion, and other purposes. The encoder and decoder are the two components of this auto encoder technology.

The input samples are typically mapped in the lower dimensional features space using a constructive feature representation during the encoding step. This procedure can be done until the necessary feature dimensional space is obtained. In the decoding step, we use reverse processing to regenerate genuine features from lower dimensional features. Figure 14 present the diagram of auto encoder.

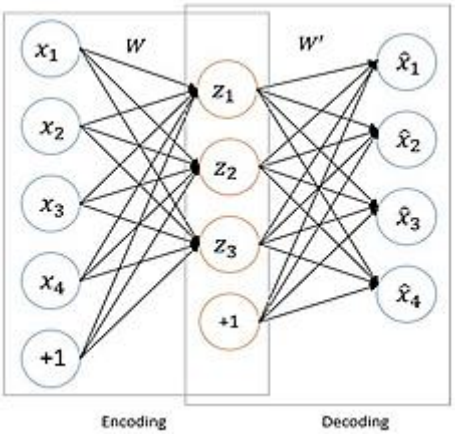


Figure 14: Diagram for Auto encoder

The encoder and decoder transition can be represented with $\emptyset$ and φ, $\emptyset : Y \rightarrow \mathscr{F}$ and $\varphi : \mathscr{F} \rightarrow Y$, then

$$\emptyset, \varphi \ = \ \text{argmik}\emptyset, \|\text{X} \ - \ (\emptyset, \varphi)\text{X}\|2 \tag{25}$$

If we consider a simple autoencoder with one hidden layer, where the input is $x \in \mathbb{R}^d = Y$, which is mapped onto $\in \mathbb{R}^p = \mathscr{F}$, it can be then expressed as follows:

$$\text{z} \ = \ \sigma 1(\text{Wx} \ + \ \text{b}) \tag{26}$$

where W is the weight matrix and b is bias. σ_1 represents an element wise activation function, such as a sigmoid or a rectified linear unit (RLU). Let us consider z is again mapped or reconstructed onto x' which is the same dimension of x. The reconstruction can be expressed a

$$x' = \sigma 2(\text{W}'z + b'). \tag{27}$$

This model is trained with minimizing the reconstruction errors, which is defined as loss function as follows:

$$(x, x') = \|x - x'\|^2 = \|x - \sigma \ (\text{W}'(\sigma \ \ (\text{W}x + b)) + b')\|2. \tag{28}$$

Usually, the feature space of $\mathscr{F}$ has lower dimensions than the input feature space Y, which can be considered as the compressed representation of the input sample. In the case of multilayer auto encoder, the same operation will be repeated as required with in the encoding and decoding phases. A deep Auto encoder is constructed by extending the encoder and decoder with multiple hidden layers. The Gradient vanishing problem is still a big issue with the deeper model of AE: The gradient becomes too small as it passes back through many layers of an AE model.

9. 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)

9.5.**Collobert and Weston (C&W) model:**

The C&W model presented in (Bhatt et al., 2017) was created using a multi-layered neural network architecture, was trained on a large dataset, and has syntactic and semantic meaning. This model is meant to be independent of any task-specific feature engineering and so functions as a viable word representation model for a wide range of NLP applications.

9.6.Word2vec:

The Word2vec model was introduced in (Mikolov et al., 2013) and is illustrated in Figure II.16 Word2vec is a shallow, two layers neural networks which is trained to reform linguistic contexts of words. This algorithm produces word vectors from an enormous amount of text of words as input by observing the contextual data the input words appear in. In the Word2vec vector space, the dimensionality of these vectors becomes usually a few hundred dimensions. In the text, every distinctive word is assigned to a related vector in the Word2vec space. In the Word2-vec space, the vectors of the words are located such that if words will occur in similar contexts, and thus, the Word2vec algorithm will discover that such words ought to be positioned in near proximity to one another in the word space. It is a mainly computationally well-organized predictive system for learning word embeddings from raw corpus. Wored2vec has two different approaches which are Skip-gram and CBOW, those two methods are algorithmically close to each other.

9.7.CBOW:

In Continuous Bag of Words (CBOW) architecture, the algorithm predicts center words(target) based on the neighboring words. It is a statistical result that Continuous Bag of Words smoothest on numerous the distributional corpus, which treats a whole context as one observation. That becomes a positive thing for small corpus.

9.8.Skip gram:

The skip gram algorithm is illustrated in Figure 15. Skip gram is very similar to CBOW, except that it exchanges the output and input. This is the inverse of Continuous Bag of Words. It predicts all surrounding words (‘‘context’’) from one input word. Essentially, the training aim of the skip gram algorithm is to discover word vectors that are useful at finding close words in the related contexts. In skip gram model, nearby context words are predicted from the center word (opposite of Continuous Bag of Words). When the corpus is bigger, skip gram model is more effective, because in skip gram model every context-center pair is treated as a new observation. The Continuous Bag of Words and skip gram architecture can be seen in Figure 15.

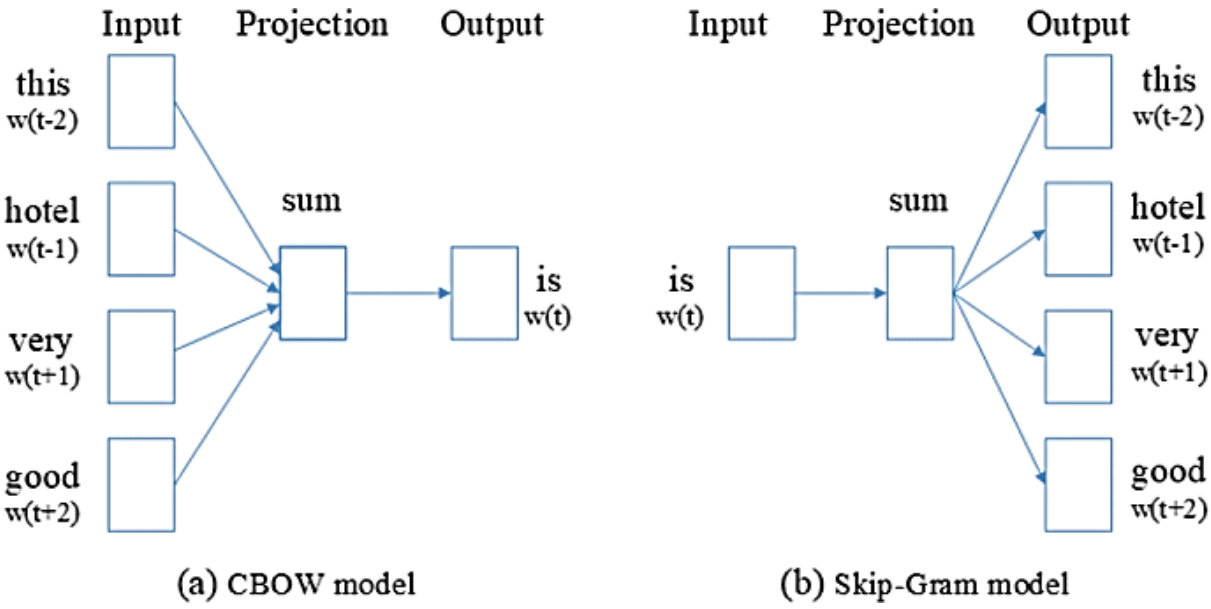


Figure 15: The skip gram and CBOW model

9.9. FastText:

Facebook AI research laboratory came up with FastText library (Joulin et al., 2017)which efficiently learns word representation. By making use of character-level information and assuming that a word is formed by n-grams of character, FastText can be used to get the representation for rare words also. Since rare words could still be broken into character ngrams, they could share these n-grams with the common words Character n-grams embeddings tend to perform superior to Word2Vec and GloVe on smaller datasets (Joulin et al., 2017). Since Word2vec and Glove consider each word as single unit and ignore the

morphological structure of the word which makes them not able to generate word embedding for the unseen or out of vocabulary words during the training. FastText overcomes this limitation by considering each word as N-gram of characters. In a way that a word vector for a word is computed from the sum of the n-gram characters when the range of N is typically 3 to 6. Consequently, FastText seems to be more effective then rest of two for social media contents which contain spelling errors and typos. (Modha & Majumder, 2019)

9.10. Global Vectors for Word Representation (GloVe):

GloVe (Pennington et al., 2014) is an unsupervised model that gives vector representations for words by using both global matrix factorization and local context window to get representation of the word (Pathak et al., 2020). More precisely, GloVe creates word embeddings by aggregating global word–word co-occurrence matrices from a corpus. Each element of word co-occurrence matrix represents how often the word i appears in context of word j. The corpus is scanned in the following manner, for each term, it looks for context terms within some area defined by a windowsize before the term and a windowsize after the term, and it gives less weight for more distant words. (Dessí et al., 2020)

9.11. Embeddings from Language Models (ELMo):

ELMo (Peters et al., 2018a)is a new type of deep contextualized word representation that model both complex characteristics of word use (e.g. syntax and semantic) and how these uses vary across linguistic contexts (i.e., to model polysemy). For example, Elmo embeddings can cleverly handle the multiple meanings of the word “stick” in the following example: Sentence 1: This stick is made up of wooden material. Sentence 2: Let’s stick to one goal at a time. Word vectors produced by ELMo are learned functions of the internal states of a deep bidirectional language model (biLM) composed of: forward language model that uses history (left context) for modeling tokens, and backward language model that inversely relies on the future (right context) for computing tokens representations. Combining these two language models to compose a bidirectional language model means that ELMo formulation have to jointly maximize the log likelihood of both the forward and backward directions as it is described in formula (29) below:

$$\sum (\log p\,(tk\,|t1, \ldots, tk-1) + \log p(tk\,|tk+1, tk+2, \ldots, tN)\ N\ k=1) \qquad (29)$$

By adding these representations to existing models, (Peters et al., 2018b) showed significant improvement in the state of the art of six challenging NLP problems, including question answering, textual entailment and sentiment analysis. They also present an analysis showing that exposing the deep internals of the pre-trained network is crucial, allowing downstream models to mix different types of semi-supervision signals.

9.11.1. ELMo Architecture:

ELMo model architecture is composed of a context insensitive type representation and 2 bidirectional LSTM layers. The context insensitive type representation uses 2048 character ngram convolutional filters followed by two highway layers and a linear projection down to a 512 representation. The 2-biLSTM layers are of 4096 units and 512 dimension projections and a residual connection from the first to second layer. The biLM provides three layers of representations for each input token, including those outside the training set due to the purely character input. In contrast, traditional word embedding methods only provide one layer of representation for tokens in a fixed vocabulary. This model was pre-trained for 10 epochs on the 1B Word Benchmark. (Peters et al., 2018b) Figure 16 shows ELMo architecture which is a bidirectional language model that combines forward (left side) and backward (right side) language models composed of 2-Layers LSTM.

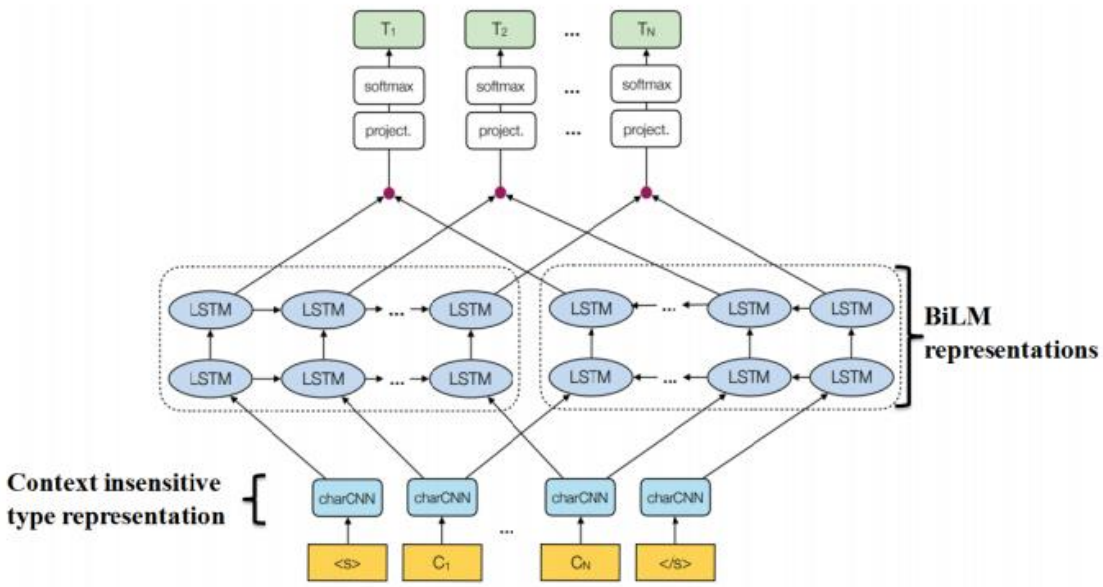


Figure 16: ELMo Architecture. (Peters et al., 2018)

9.12. Graphs from LOw-level unit Modeling (GLoMo):

Unsupervised latent graph learning is used to generate graphs from the low-level unit modeling (GLoMo) framework (Yang et al., 2018). It is also a transfer learning framework designed to increase the performance of natural language processing tasks such as sentiment analysis, natural language inference, question answering, and picture classification.

9.13. Universal Language Model Fine-tuning (ULMFiT):

The ULMFiT (Howard & Ruder, 2018) transfer learning paradigm may be applied to any natural language processing problem. ULMFiT's pre-trained models may be used for sentiment analysis. A language model is pre-trained on the general domain and then fine-tuned on the target domain in this method.

Its operation is independent of document size, quantity, and label, and so claims to be universal. It uses a single architecture and training to perform a variety of activities and does not require domain-specific documents or labels.

9.14. Bidirectional BERTEncoder Representation from Transformers (BERT):

This section introduces BERT and its implementation in detail. BERT is a Google-created language representation model. It is the first deep bidirectional language representation model that combines left and right context conditioning in all layers to pre-train deep bidirectional representations from unlabeled text. Thus, with only one additional output layer, the pre-trained BERT model may be fine-tuned to provide state-of-the-art models for a wide range of NLP tasks, including question answering and language inference, without requiring significant task-specific architectural changes. (Devlin et al., 2019a)

9.14.1. Model Architecture:

BERT model architecture is a multi-layer bidirectional transformer encoder based on the original implementation described in (Vaswani et al., 2017) and released in the tensor2tensor library11. By using TPUs put together in a V3 pod, (Devlin et al., 2019b)build two different sizes models: BERTBASE and BERTLARGE. BERTBASE is composed of 12 transformer blocks with 12 self-attention heads. This model of 110M parameters has a hidden size of 768. BERTLARGE is built in 24 layers with 16 self-attention heads having total of 340M parameters and 1024 as hidden size. For comparing between bidirectional self-attention carried out by BERTBASE model and constrained self-attention of OpenAI GPT model (Radford et al., 2018) the authors chose to build BERTBASE for having the same model size as OpenAI GPT. Figure 17 below provides information about the architecture of BERT models proposed in (Devlin et al., 2019b)contribution.

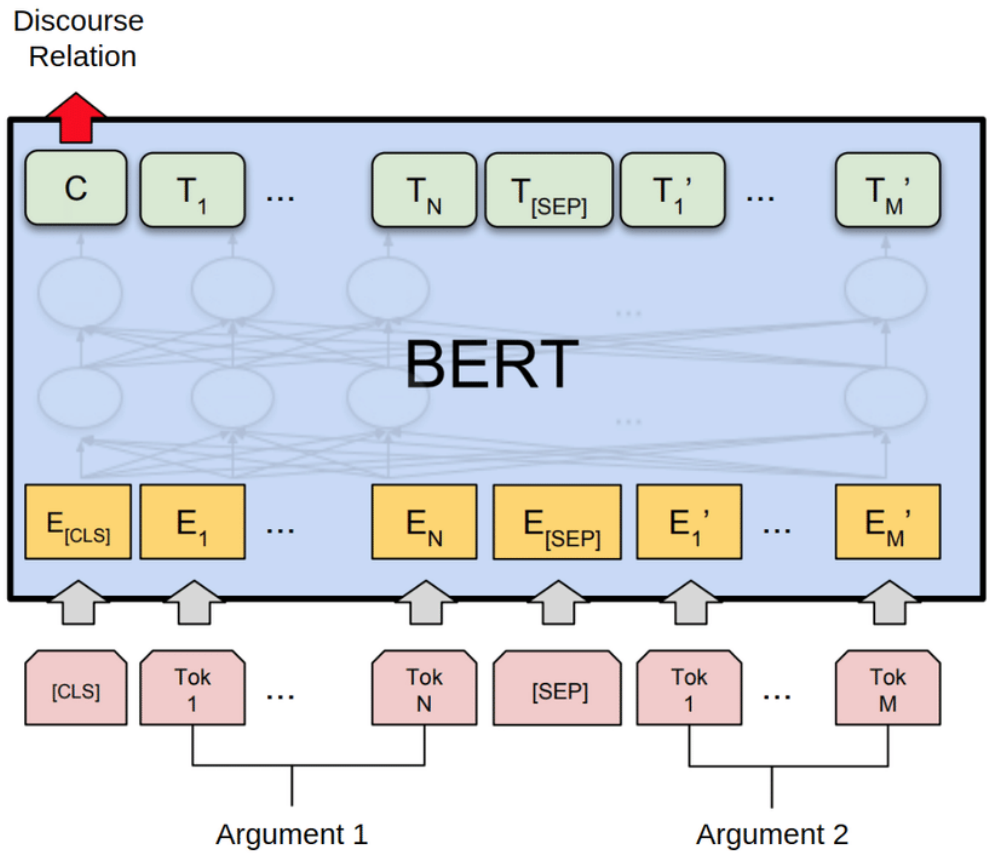


Figure 17: The architecture from BERT (Devlin et al., 2019)

9.14.2. BERT Input Representation:

A "sequence" refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together (e.g., for question answering task), where a "sentence" can be an arbitrary span of contiguous text of maximum 512 tokens, rather than an actual linguistic sentence. This enables BERT model to handle a variety of down-stream tasks. (Devlin et al., 2019b) use WordPiece embedding (Wu et al., 2016a) of 30,000 token vocabularies for representing input tokens. The first token of every sequence should be the special BERT classification token [CLS]. The final hidden state of this token is the aggregate sequence representation used in BERT fine tuning for classification tasks. Whereas for tasks requiring sentence pairs as input, BERT encoding proposes packing the sentence pairs together into a single sequence, and differentiates between these two sentences by two ways. First, it separates them with a special token [SEP]. Second, it adds a learned embedding to every token indicating whether it belongs to sentence A or sentence B. BERT input representation is constructed by summing the corresponding token, segment, and position embedding. A visualization of this construction can be seen in Figure 18.

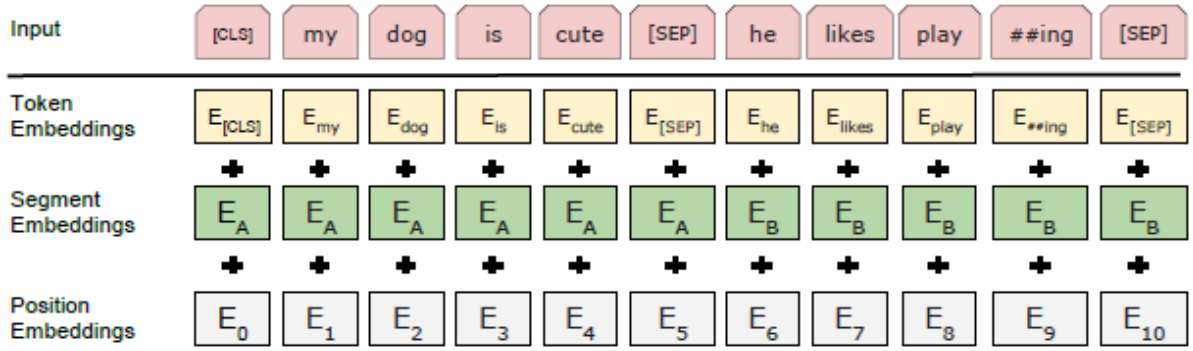


Figure 18: BERT input representation

9.14.3. BERT Output Representation:

Two kinds of outputs are described in (Devlin et al., 2019b)paper, The first one is the vector representation of the special token [CLS] which is the final hidden state corresponding to this token and it is used as the aggregate sequence representation for classification tasks (Figure 19 left side). Then, the second kind of outputs refers to the token final hidden states (Figure 19 right side). In the figure below, the outputs of BERTBASE model are illustrated where input embedding is denoted as E, the final hidden vector of the special [CLS] token as C ∈ $\mathbb{R}H$, L is the number of layers composing BERT model (. i.e., 12 for BERTBASE and 24 for BERTLARGE), and the final hidden state for the i th input token as Ti ∈ $\mathbb{R}H$ with H refers is the hidden size of BERT (. i.e., 768). The left side of this figure shows the aggregate sequence BERT representation and the right side represents the tokens BERT representation.

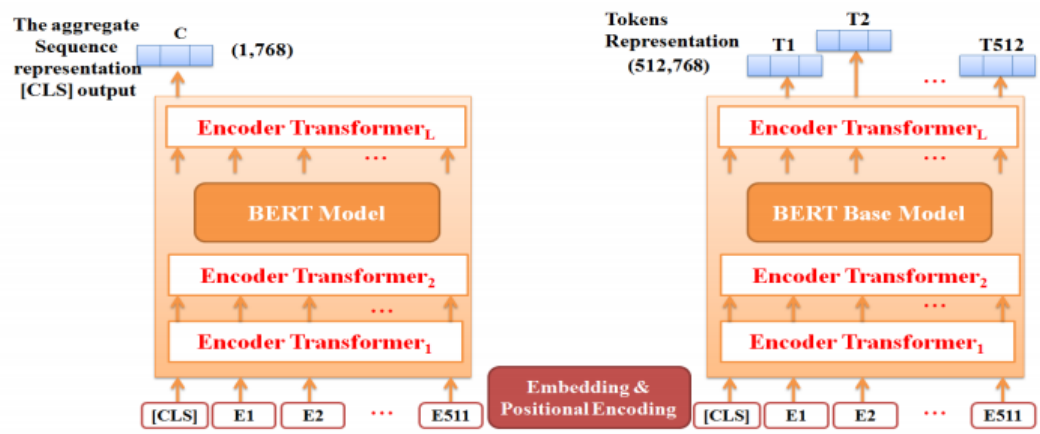


Figure 19: BERT outputs representation

Part2:

1. Related work:

This section provides a detailed overview of the important works and extant research in the field of automated detection and hate speech in particular.

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

Several studies have looked into many comparable terminologies that provide comparable notions to the phenomenon of hate speech (e.g., cyber-bullying, abusive language, radicalization detection). The examination of these many terminologies will undoubtedly aid in gaining insights from many viewpoints in current events, as well as finding and understanding the interrelationships between them.

1.2. Abusive language detection:

It is a broad notion 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 inappropriate language, (Y. Chen et al., 2012) utilized YouTube comments as a dataset. They used a combination of lexical and syntactic features and they incorporated the user's writing style to predict the user's behavior in the future. Also, they assumed that they would be able to distinguish abusive phrases from negative polar expressions. They used a basic lexicon to take a small selection of negative polar terms and then crowdsourced the rest, the abusive words were labeled. (Xiang et al., 2012) suggested a similar methodology to detect offensive content on Twitter. Their characteristics were based mostly on the linguistic regularities of profane phrases, as well as statistical topic modeling on a large dataset. In the case of a deep learning situation, using the dataset supplied by (Waseem & Hovy, 2016), (Park & Fung, 2017) examined the performance of one-step and two-step classifiers. Based on their results, they believe that combining 2 classifiers (e.g. CNN and logistic regression) can boost up the performance. Moreover, FastText was utilized by (Y. Chen et al., 2012) as a neural network classifier to detect abusive content on multiple social media sites. They discovered that the performance of FastText is inferior to that of SVM as a classifier.

1.3. Cyber bullying detection:

(Dinakar et al., 2011) Research is one of the pioneers and most cited researches for textual cyberbullying detection. Their experiment was based on a corpus of 4500 YouTube comments. Their findings revealed that displaying the dataset's polarities outperformed classifying the dataset into many classes. Both of (Nahar et al., 2014) and (Capua et al., 2016) presented an unsupervised approach for cyber bullying detection.

The study of (Özel et al., 2017) is unusual in this field since they looked into Turkish language in order to identify cyber bullying from text on Twitter and Instagram. Their results showed that Naïve Bayes Multinational showed the best results in both accuracy and total training and testing time. Finally, for cyber bullying detection, (Pawar et al., 2018) used distributed computing. Their study is primarily concerned with the robustness of the performance rather than the accuracy alone.

1.4. Hate speech detection:

(Warner & Hirschberg, 2012) were among the first to automate the identification of hate speech on the Internet, starting in the early stages. Their work involved a number of problems that should have been resolved by now. Their research was expressly designed to identify anti-Semitism. The first racial oriented research was by (Huang et al., 2018) who decided to follow (Warner & Hirschberg, 2012) path and implemented a supervised model to detect racist tweets. Then, (Burnap & Williams, 2014) were motivated to investigate the spread of hate speech immediately after Lee Rigby murder in UK. They trained a supervised classifier to find hateful and non-hateful tweets related to this particular event. (Waseem & Hovy, 2016) served as a foundation for several other studies that looked at predictive factors for hate speech identification. They gave us access to their massive 16K twitter corpus, which is dedicated to hate speech study in English.

It was also looked at hate speech in various languages. (Del Vigna et al., 2017) are working on a model to identify hate speech in Italian on Facebook. The classifiers were unable to distinguish between three levels of hatred, according to their findings. In addition, (Alfina et al., 2017) tested the capacity to identify hatred in Indonesian language. They used a political event to get their sample "Jakarta Governor election 2017."Because of the German anti-Semitism (Jaki & De Smedt, 2019) proposed a technique to identify hate speech in German language as a swift response to the latest German NetDG regulation. From a political communication standpoint, he also produced a complete qualitative and quantitative examination of what constitutes hate speech. Recent work has switched to using deep learning for such tasks; for example, (Gambäck & Sikdar, 2017) used deep learning on Waseem and Hovy's dataset (Waseem & Hovy, 2016). Precision and recall were exceeded by their outcomes. (Joulin et al., 2017) utilized the same corpus to compare different deep learning model combinations. (Zhang et al., 2018) also looked at integrating convolutional and gated recurrent unit networks, as well as comparing their results to all existing deep learning models. They stated that their work sets a new benchmark for future researches in this area. Finally, (Pitsilis et al., 2018) believe that deep neural networks have a high potential to solve the issue of hate speech detection. Their deep learning approach outperformed all the state-of-art approaches.

1.5. Arabic Hate Speech Detection:

Only a few Arabic researchers have made significant contributions to this field. On the other side, numerous Arabic studies have been conducted in related areas that we might refer to as “anti-social behaviors,” such as abusive or insulting language and cyber bullying.

1.5.1. Arabic anti-social behavior detection:

Begin with the detection of abusive language. Abozinadah set the path in this field, contributing to three studies specifically designed for it. First, (Abozinadah et al., 2015) suggested a strategy to Arab governments in response to their requirement to restrict such offensive materials. They developed their own test set and made it available to the public.

Then, in (A. Abozinadah & H. Jones, Jr, 2016), they improved on the prior work by introducing a lexicon supplied by an Arabic word correction approach to improve the identification of offensive terms. Abozinadah's third study is (Abozinadah & Jones, 2017), which employed a statistical learning strategy for detection to address the limitations of the BOW technique demonstrated in prior publications. (Mubarak et al., 2017)'s effort aims to create a huge corpus of Arabic tweets categorized as (obscene, offensive, and clean) and make it available to future scholars.

Alakrot et al. contributed two more papers (Alakrot et al., 2018a) (Alakrot et al., 2018b). In the first piece, they built a corpus of Arabic comments from YouTube and made it publicly available for abuse detection. An empirical evaluation of the dataset was undertaken in their second article. They came to the conclusion that combining N-gram and stemming may result in poor performance. (Lefever et al., 2018) followed in the footsteps of Abozinadah and Azalden, however their notion differed slightly. They produced a big amount of pornographic material on Arabic Twitter. As a result, a vast vocabulary was created based on that corpus.

In addition to the above mentioned behaviors, cyberbullying is a severe issue that has been addressed by several English-language scholars. (Haidar et al., 2017) were the first to attempt to detect cyber bullying in the Arabic language. Their study was the first step in this direction, since it still requires several improvements, such as taking into account additional cyberbullying-related variables and selecting better feature representation.

In Arabic, (Alduailej & Khan, 2017) outlined the key obstacles of recognizing cyberbullying. The underlying difficulty was that we needed to determine the context before evaluating whether or not it was cyberbullying.

1.5.2. 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.

1.5.3. 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.

1.6. Summary and Analysis

The tables that follow provide a summary of all the articles that have been discussed, and they are sorted by time series. These tables address the following subjects in turn: Antisocial behavior in English, hate speech in English, and finally, antisocial behavior in Arabic. These tables may be used as a fast reference for all of the important work done in social media automated detection. All of the methodologies, as well as the outcomes of their various studies, are summarized concisely. Table 3,4,5 compiles all of the hate speech terms and their respective contributions. Table 6 includes all multilingual contributions and publications dealing with hate speech.

Finally, table 7 emphasizes the Arabic language by collecting all publications dealing with the identification of anti-social behavior in social media platforms. Table 8 show the Algerian dialect contributions. The best results from each study are highlighted in the results column.

Table 3: 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 4: Summary of the current state of cyberbullying detection, and their respective results, in the metric: Precision (P), Recall (R), F1-Score (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
(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 5: 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	Classes	ML Approach	Features Representation	Algorithm	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
(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 6: Summary of the Italian,Indonesia and German contributions in anti-social behavior detection results, in the metrics: Precision (P), Recall (R), F1-Score (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	0.75	0.68	0.71
					RNN and LSTM	0.70	0.75	0.72
General hate speech (Indonesian)								
(Alfina et al., 2017)	2017-Twitter	Hate speech /Non hate speech	Supervised	BOW and n-gram	RandomForest decision tree	-	-	0.93

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

Author	**Year-Platform**	**Classes**	**ML Approach**	**Features Representation**	**Algorithm**	**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
(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, Regular user	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 SentiStrengthFeature 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
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

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

Table 8: Summary of the Algerian dialect contributions in hate speech detection and their respective results, in the metrics: Precision (P), Recall (R), F1-Score (F).

Conclusion

In this chapter, we discussed the various deep learning algorithms for hate speech, as well as the strategies used by each method. Following that, we showed some deep learning research and studies for both English and Arabic. Finally, we discussed similar work that uses deep learning models to analyze hate speech in Algerian dialect.

CHAPTER 03:
PROPOSED APPROACH FOR ALGERIAN HATE SPEECH DETECTION

Table of Contents

Chapter 03: Proposed approach for Algerian hate speech detection 57
Introduction 57
1. Global Architecture 57
2. Data creation: 59
2.1. Data collection: 59
2.2. Retrieving methods: 59
2.2.1. Facebook API (application programming interface) 59
2.2.2. Twitter API (application programming interface) 60
2.2.3. YouTube API (application programming interface) 60
2.3. Saving comments 60
2.4. Comments filtering 60
2.5. Comments annotation 61
3. Data preprocessing 61
3.1. Cleaning comments 61
3.2. For the deep learning basic models 62
3.2.1. Tokenization 62
3.2.2. pad sequence to a specific length 62
3.3. For transfer learning models 62
3.3.1. Tokenization: 62
3.3.2. pad sequence to a specific length 62
3.3.3. Creating attention masks for sentences tokens 62
4. Features extraction 63
5. Classification 63
5.1. Machine learning 63
5.2. deep learning 63
5.2.1. LSTM, GRU, Bi-LSTM and Bi-GRU 63
5.2.2. CNN 64
5.3. Transfer learning 65
5.3.1. Bert model 65
5.3.1.1. Bert embedding +LSTM 68
5.3.1.2. Bert embedding + GRU 68
5.3.1.3. Bert embedding and GRU+LSTM 69
5.4. Ensemble learning 70
6. Evaluation Measures 73
Conclusion 73

Introduction

In this chapter, we present a new approach to hate speech in Algerian dialect, beginning with the general architecture and describing each step in detail. In this approach, we collect a corpus that will go through many phases of pre-processing for both deep learning and transfer learning models, in addition to presenting a deep learning architecture that will allow us to train a model.

1. Global Architecture

In order to develop an intelligent approach for hate speech detection, we present here our global architecture, explaining our general approaches, and its fundamental phases.

First of all, we create a dataset that contain a set of comments and labels. The second procedure we complete is preprocessing, for which we provide two primary preparation views based on the models into which our data will be fed. One preprocessing viewpoint is dedicated for pretrained language models such as (Bert-base-arabic, mBert, and mDistilBert), while the other preprocessing perspective is dedicated for deep learning models. On the one hand, we feed our preprocessed comments into pretrained language models. On the other hand, in order to train deep learning, we convert preprocessed comments into vectors. We get these word representations by either training the FastText algorithm on our comments or by using existing pretrained word vectors. Figure 20 shows the general steps for constructing the annotated corpus starting with the collection and the filtering of comments then the annotation step and finally the pre-processing phase. While figure 21 shows the different algorithm for feature extraction with both ML and DL model. Figure 22 depicts the general architecture of our suggested transfer learning models.

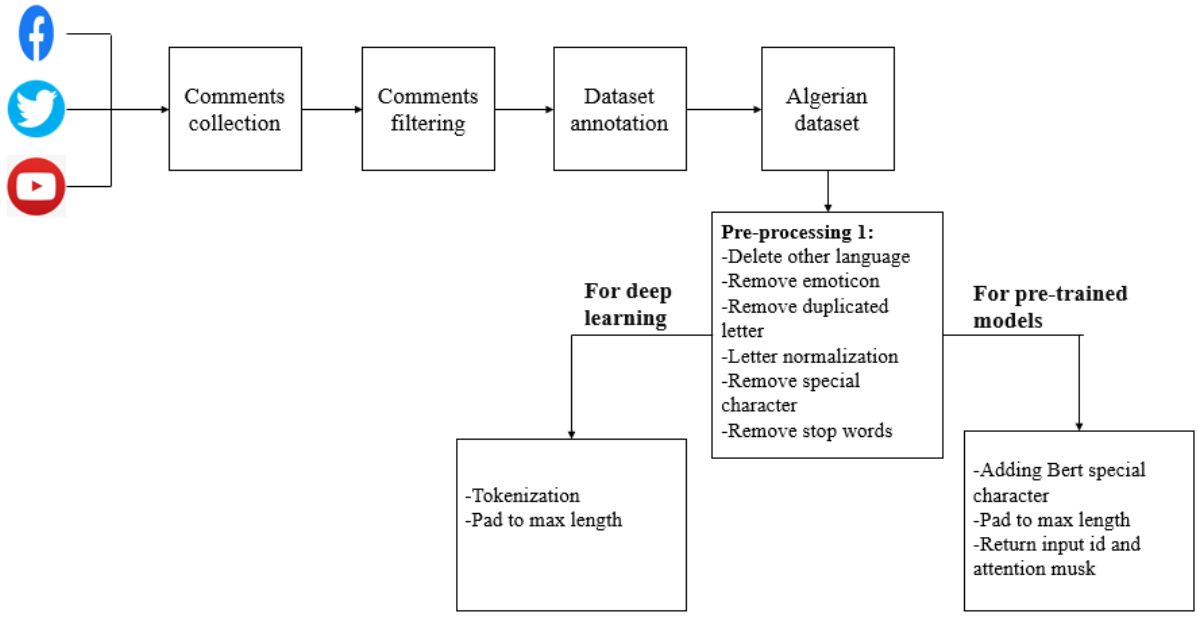


Figure 20: General steps for constructing the annotated dataset

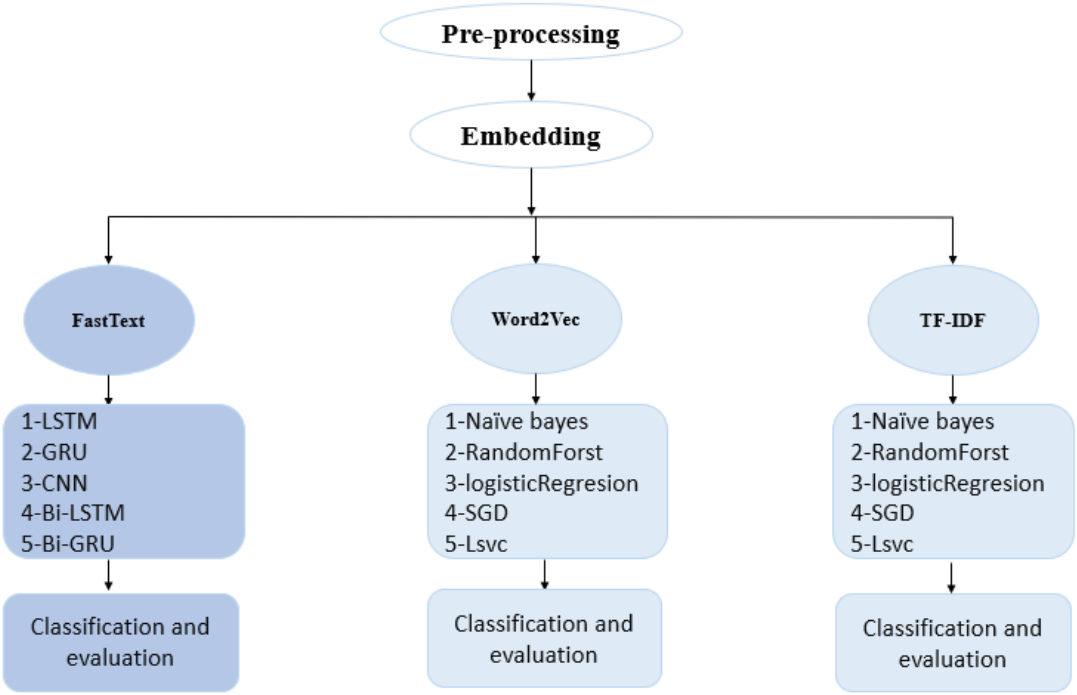


Figure 21: DL and ML classification

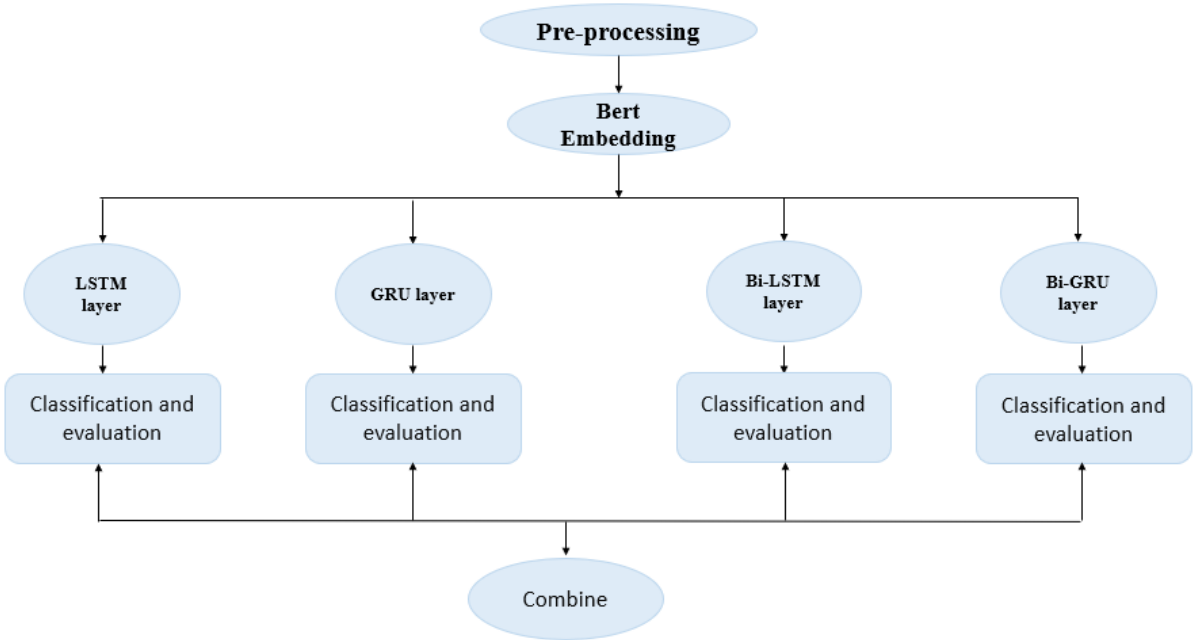


Figure 22: Global overview architecture

2. Data creation:

Primarily, the process of preparing the data is described in Fig 23. It starts by collecting the comments, annotating the data, cleaning the data from any irrelevant or redundant comments, normalizing the comments, tokenizing, and text vectorization for the objective of feature representation.

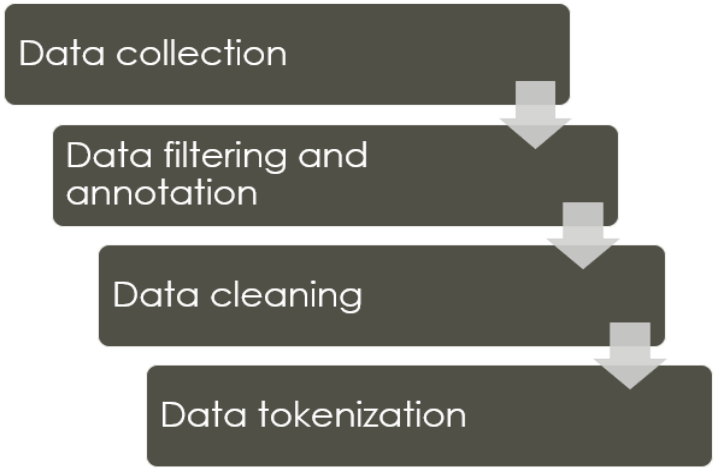


Figure 23: Steps of data collection and preprocessing.

2.1. Data collection:

Our dataset consists of comments collected from 3 social media platforms: Facebook, YouTube and Twitter. We collected 14150 comments from various popular Algeria pages and accounts, including Echourouk news TV, El Djazairia, Abdelmadjid Tebboune, and others. These comments are about 12 different topics of hate speech and cyberbullying, as well as offensive and abusive language, such as religion, parliamentary politics, Moroccan politics, misogyny, and so on. These comments are saved in a xlsx file.

2.2. Retrieving methods:

We chose the following automated approaches since manually retrieving comments is a difficult procedure.

2.2.1. Facebook API (application programming interface)

Facebook graphAPI[11] is a service that each Facebook user can use. We started by creating a Facebook developer account, then an app on their platform, and finally a Python script that allows us to access Facebook's API to extract posts and comments.

[11] API Graph (facebook.com)

2.2.2. Twitter API[12] (application programming interface)

Twitter API is also a platform that every user with a Twitter account can use by creating applications on the web platform. then creating a python script to connect to the Twitter server and extract the tweets.

2.2.3. YouTube API[13] (application programming interface)

For Google API YouTube, a user with a Gmail account can use the services. We created access codes to access them like for Facebook and Twitter. and then a python script to extract and collect comments from YouTube videos, then saving them in a csv file.

2.3. Saving comments

The APIs of the three social networks send numerous information while obtaining messages. We keep only the messages which are saved in the dataset's comment column, and the source of the comments in where we retrieved the comments.

2.4. Comments filtering

The retrieved comments were all filtered before saving in the dataset as follow:

- We manually deleted all the comments that were written in different languages from French /English and Arabic or Algerian dialect.
- All the retweets of a given tweet are removed and all the repeated comments are also removed.
- The identical comments that contain the same contents are filtered to ensure that the text of each considered comment is unique.
- The very long comments that contain more than 50 words were filtered and the very short comments that contain under 10 words are also filtered.

After this filtering step we have ended up with a total of 14150 comments. Figure 24 shows some information about the dataset

[12] Twitter API Documentation | Docs | Twitter Developer Platform
[13] YouTube Data API | Google Developers

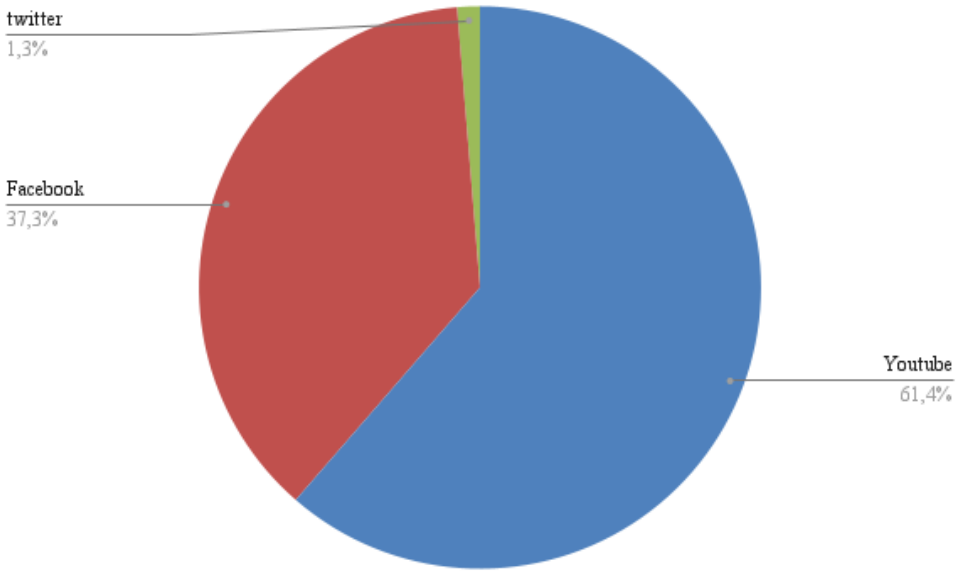


Figure 24: Information about the collected dataset.

2.5. Comments annotation

Following the extraction and filtering phases, we moved to the manual annotation stage. The annotation was done by three annotators using three labels (hate speech, cyberbullying, abusive and offensive language), there are two classes for each label. yes for (hate/ bullying / abusive and offensive) and no for (non-hate/ non-bullying / non-abusive and offensive). Table 9: shows some information about the labels.

Table 9: Information about the dataset

Hate speech		**cyberbullying**		**abusive and offensive language**	
Yes	**No**	**Yes**	**No**	**Yes**	**No**
7573	6577	4525	9625	5766	8384

3. Data preprocessing

After the annotation step, we must go through many rounds of preprocessing which is critical for obtaining a practical corpus. In the next paragraphs, we shall discuss them.

3.1. Cleaning comments

- First, we removed all the language except Arabic.
- We filtered all the emoticons.
- Replace the duplicate letters with a single instance of the letter.

- Letter normalization: unifying the letters that appear in different forms. Example we replace {أ إ ؤ ء آ ئ} with {ا}. Also remove all of the Arabic diacritics 'Tash keel' like (tashdid, Fatha, kesra …etc.) and the elongation as (اجـــــــــــراءاتـــــ become اجراءات).
- All non-Arabic characters, numbers, symbols, punctuation, hashtags, web addresses, diacritics are removed.
- Then we filtered both the Arabic and Darija stop words.
- Finally, deleting all the empty lines.

3.2. For the deep learning basic models

3.2.1. Tokenization

Now that we have cleaned data that contains only significant words, we can pass into the tokenization process to divide the text into separated words called tokens. The very simplest method for tokenizing text is to split on whitespace.

3.2.2. pad sequence to a specific length

Pad sequences is used to ensure that all sequences in a list have the same length. By default, this is done by padding 0 in the beginning of each sequence until each sequence has the same length as the longest sequence.

3.3. For transfer learning models

3.3.1. Tokenization:

Add [CLS] at the beginning and [SEP] at the end of the sentence. The role of these tokens is to define the boundaries of the sentence, Tokenizing sentences to word pieces by using a transformer tokenizer that follows WordPiece tokenization (Wu et al., 2016b).

3.3.2. pad sequence to a specific length

Truncate longer sentences and pad shorter ones to the max length which is 100 in our case. Encoding (embedding) sequences tokens by embedding of 23378 tokens vocabulary size. This encoding step results an array named Inputs IDs.

3.3.3. Creating attention masks for sentences tokens

This attention mask encoding allows differentiating between padded and non-padded tokens in a way that padded tokens will not be included in the attention vectors calculations.

4. Features extraction

We use three different algorithms for feature extraction which are, Word2vec (Mikolov et al., 2013), FastText (Joulin et al., 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).

5. Classification

For comparing the results, we use both classification methods, classic and deep learning based.

5.1. Machine learning

For classic method, we use both TF-IDF and Word2Vec with five classification Algorithms such as: RandomForset (RF), multinomial naive bayes, LogisticRegression, SGDClassifier) and LinearSVC). For their implementation phase, we were inspired by the classification algorithm proposed by (Joshi et al.,2016).

5.2. deep learning

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

5.2.1. LSTM, GRU, Bi-LSTM and Bi-GRU

For LSTM, GRU, Bi-LSTM or Bi-GRU, the first layer is a randomly-initialized word embedding. The weights of the embedding matrix are calculated using FastText. then the (LSTM/GRU/ Bi-LSTM / Bi-GRU) layer that scans the feature map. For reducing over-fitting, a Dropout layer is added. The obtained scores are then feed to a single feed-forward (fully-connected) layer, the output of that layer goes through a sigmoid layer that predicts the output classes. Figure 25 shows the architecture

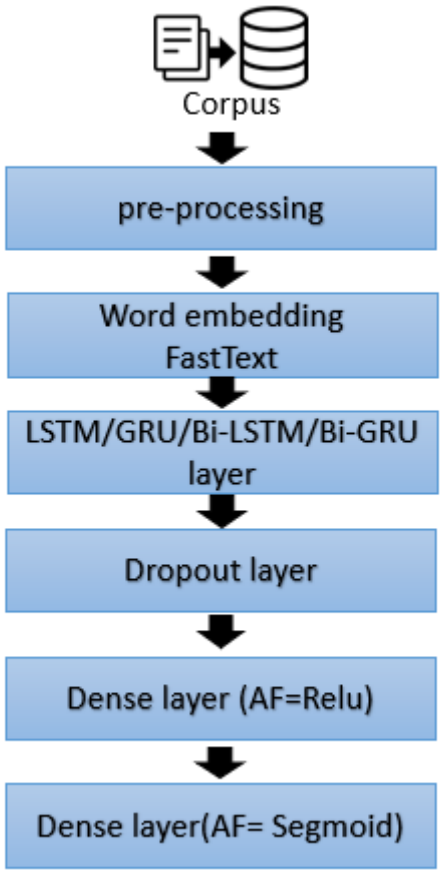


Figure 25: DL architecture

5.2.2. CNN

For CNN, the first layer is a randomly-initialized word embedding layer that turns words in sentences into a feature map. then the CNN layer that scans the feature map. Global maxpooling is applied to the output layer. The obtained scores are then feed to a single feed-forward (fully-connected) layer, the output of that layer goes through a sigmoid layer that predicts the output classes. Figure 26 shows the architecture

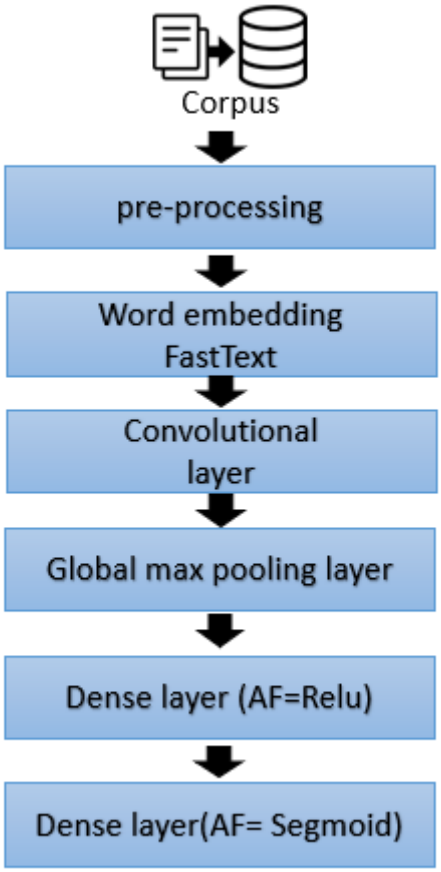


Figure 26: CNN architecture

5.3. Transfer learning

Pretrained transformer models have been recently used in many NLP tasks and have continuously achieved new state-of-the-art results(Guellil et al., 2018).

5.3.1. Bert model

In our experiments we use three pretrained Bert models:

AraBERT(Guellil et al., 2018)**:** Arabic Pretrained BERT (Bidirectional Encoder Representations from Transformers) base language model (Guellil et al., 2018) pretrained on on ~8.2 Billion words gathered from Arabic version of OSCAR (unshuffled version of the corpus) - filtered from Common Crawl and Recent dump of Arabic Wikipedia and other Arabic resources which sum up to ~95GB of text.

Multilingual BERT (mBERT):

Pretrained model on the top 102 languages with the largest Wikipedia using a masked language modeling (MLM) objective. It was introduced in(Abozinadah et al., 2015). This model is uncased: it does not make a difference between english and English.

Distil Bert Multilingual[14]: The model is trained on the concatenation of Wikipedia in 104 different languages. The model has 6 layers, 768 dimension and 12 heads, totalizing 134M parameters (compared to 177M parameters for mBERT-base). On average DistilmBERT is twice as fast as mBERT-base.

The three aforementioned transformer models have been trained solely on modern standard Arabic. Thus, they may have difficulty dealing with Algerian dialects. For this reason, we performed further pretraining (or fine-tuning) for AraBERT, mBERT and distilmBERT models using our corpus. This is done to make the transformer language models more familiar with its vocabulary.

We add to BERT model an additional output layer having the aggregate sequence representation (i.e. [CLS] output) as input. This output layer is simply a dense layer having the number of classes as units (i.e., classification layer). Then, we train the resulted araBERTbase model on the training set for 5 epochs. We use the value of 2e-5 as learning rate (. i.e. lr = 0,00002) with Adam optimizer as it is suggested by(Devlin et al., 2018; Mozafari Marzieh and Farahbakhsh, 2020; Sohn & Lee, 2019). Other than these hyper parameters choices, we keep the model parameters as default. Figure 27 shows BERT architecture

[14] distilbert-base-multilingual-cased · Hugging Face

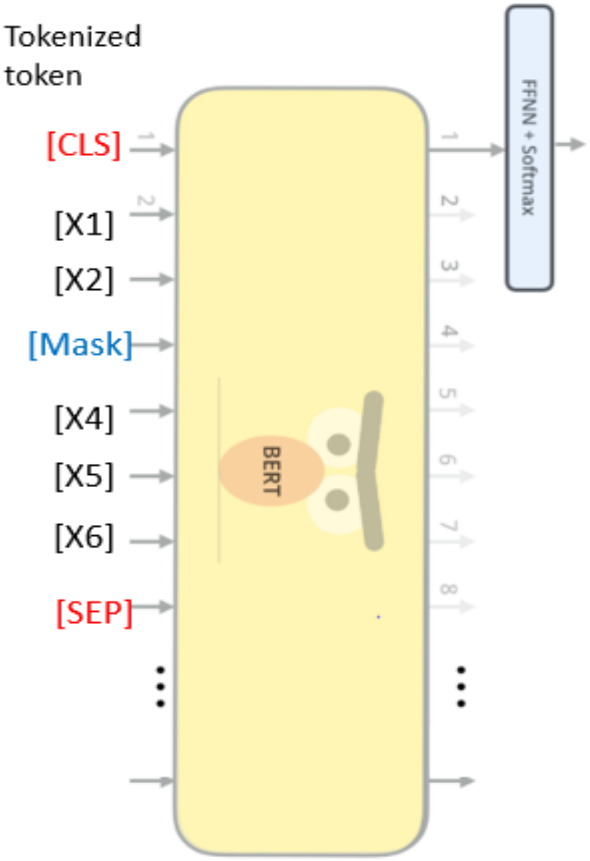


Figure 27: BERT architecture

We also have used ANN, RNN, Bidirectional RNN Dense layers with the BERT model.as it is showing in figure 28. The number of units chosen for each of the layers was based on the experimentations carried out. The loss function and activation function used were cross-entropy loss and sigmoid activation for all the models.

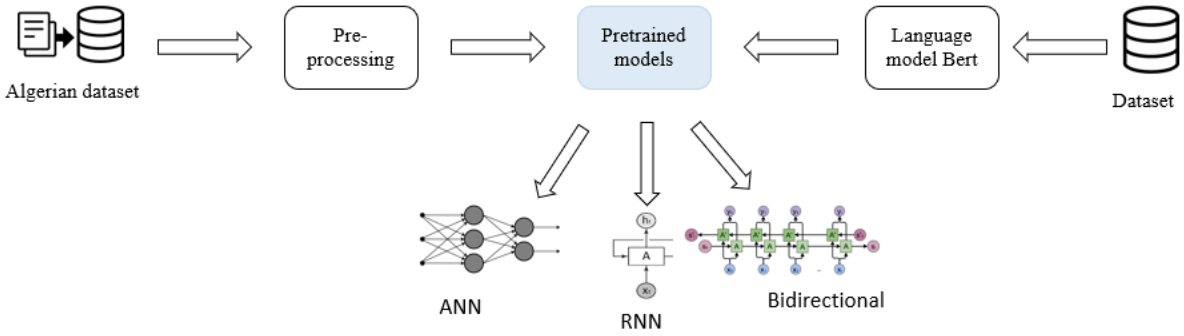


Figure 28: BERT and DL layer architecture

5.3.1.1.Bert embedding +LSTM

In this model, we choose to add LSTMs to the BERT model and make the final BERT outputs of tokens as their inputs. We prefer to use two LSTMs rather than using only one layer and then concatenate them and pass it into the dropout layer and then to the last dense layer with Softmax activation function. The architecture is descripted in figure 29.

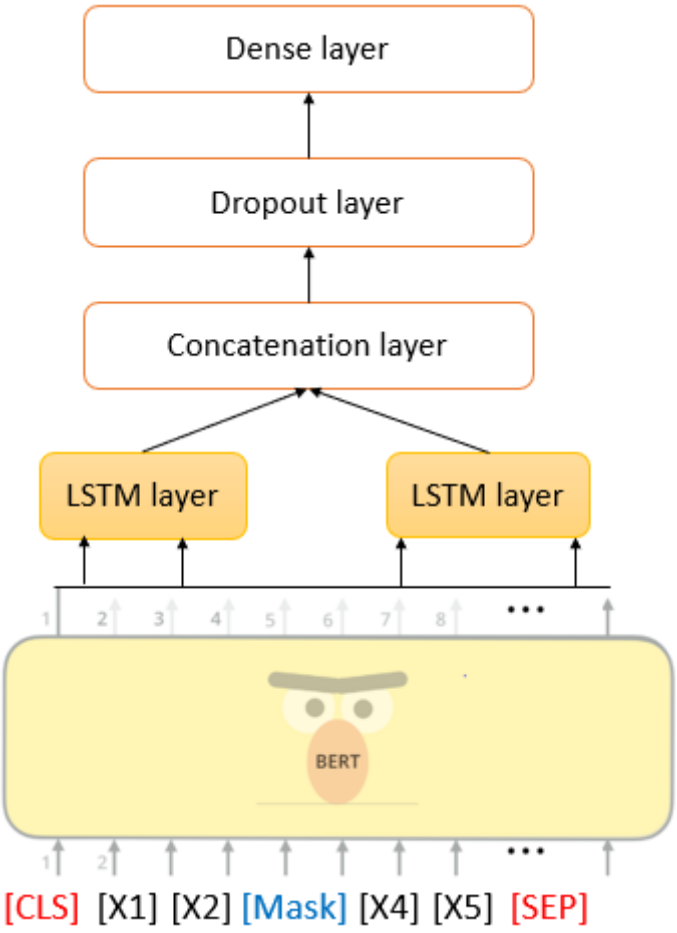


Figure 29: Bert layer with LSTM layer

5.3.1.2.Bert embedding + GRU

In this model, we choose to add LSTMs to the BERT model and make the final BERT outputs of tokens as their inputs. We prefer to use two LSTMs rather than using only one layer and then concatenate them and pass it into the dropout layer and then to the last dense layer with Softmax activation function. Figure 30 descript the architecture.

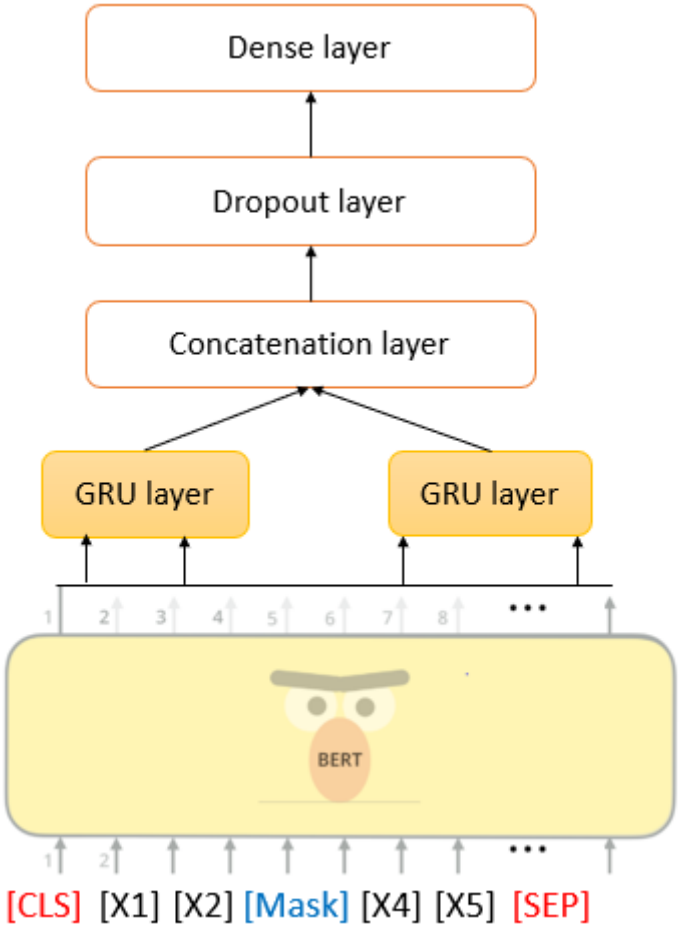


Figure 30: Bert layer with GRU

5.3.1.3.Bert embedding and GRU+LSTM

In this model, we choose to add LSTMs and GRUs to the BERT model and make the final BERT outputs of tokens as their inputs. We prefer to use one LSTMs layer and one Grus layer and then concatenate them and pass it into the dropout layer and then to the last dense layer with Softmax activation function. Figure 31 shows the architecture.

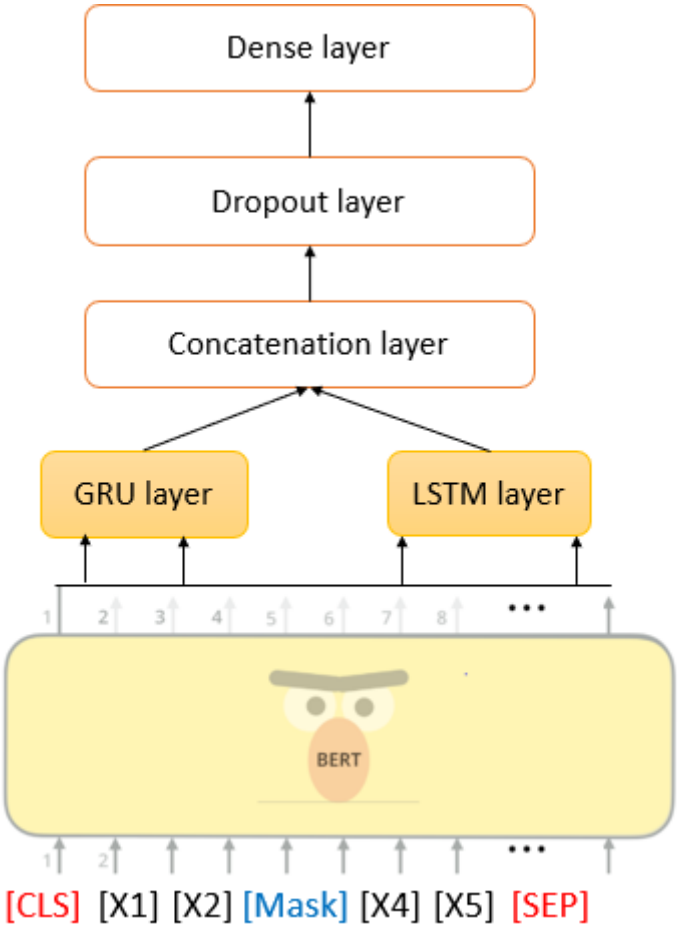


Figure 31: Bert layer with both GRU and LSTM

5.4. Ensemble learning

For ensemble learning, we combine the models created in the previous section and train each one of them independently on the same data.

- The first model is Bert and LSTM with Bert embedding. figure 32

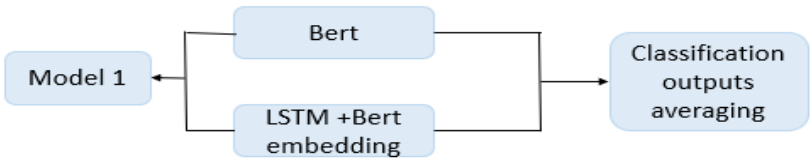


Figure 32: Bert and LSTM with Bert embedding

- The second one is Bert and Gru with Bert embedding. Figure 33

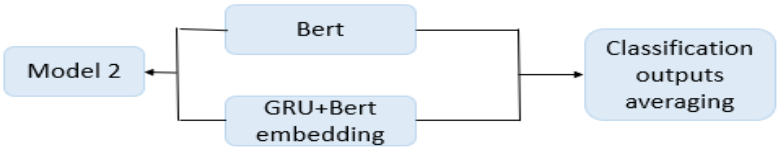


Figure 33: Bert and GRU with Bert embedding

- The third one is Bert and both (LSTM and GRU) with Bert embedding. Figure 34

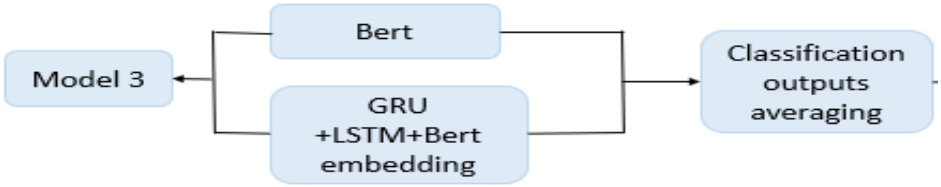


Figure 34: Bert and both GRU and LSTM with Bert embedding

And then, we combine between the first and the second model, the first and the third model, the second with the third and finally between all the three models. for producing the prediction of the ensemble model for one given comment, we combine the trained model's outputs for this comment by calculating their mean. The new prediction resulted after average calculation is considered the prediction of the ensemble learning model for this comment.

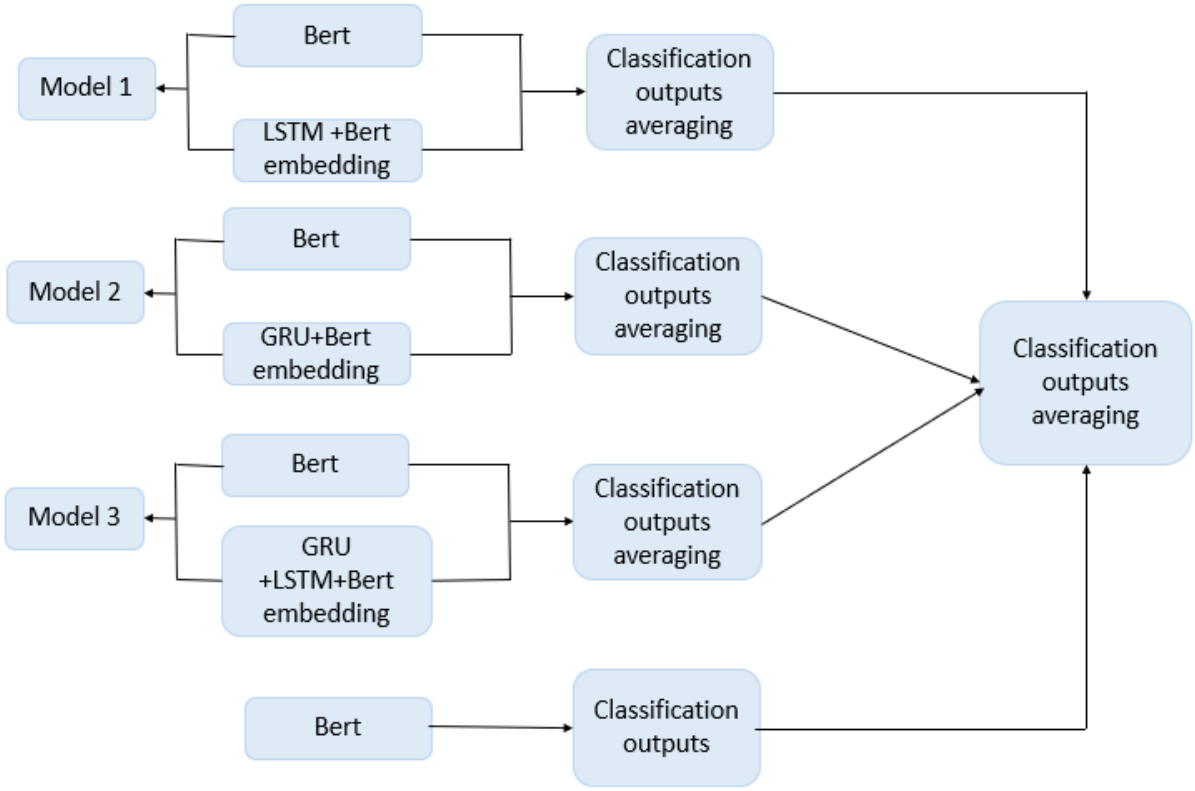


Figure 35: Ensemble learning architecture

The formula 30 shows the average prediction calculation rules.

$$\text{AveragePrediction} = \frac{1}{M}\sum_{i=1}^{pi} mi \qquad (30)$$

Where Pi denotes the prediction of the $i\ th$ model, and M is the number of models composing the ensemble models

We suppose that training different deep learning architectures separately enables us to avoid overfitting locally (i.e., in each model apart). Otherwise, we find that an ensemble learning strategy for fine tuning BERT is better than adding layers that builds more complex models. Since complexity can lead to overfitting and requires more RAM space for execution.

6. Evaluation Measures

The used performance evaluation measures are accuracy, precision, and recall of both pretrained language model and models trained from scratch, we use f1_score and accuracy as evaluation and validation metrics for our approaches.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (31)$$

$$\text{Precision} = \frac{TP}{FP+TP} \quad (32)$$

$$\text{Recall} = \frac{TP}{FN+TP} \quad (33)$$

$$\text{F1} = 2 \text{ x Precision x Recall / Precision} + \text{Recall} \quad (34)$$

Where TP is the true positives, TN is the true negatives, FP is the false positives and FN is the false negatives.

Conclusion

We introduce in this third chapter the different parts we accomplish for Algerian dialect. Starting by the preprocessing phase, which includes two different kinds of preprocessing depending on the model that the preprocessed comment will be fed in. Then presenting the main proposed approaches to deal with hate speech detection for Algerian dialect, which are machine learning classifiers, deep learning classifiers, fine tuning pretrained language models and creating a deep learning architecture.

CHAPTER IV:
TEST AND RESULTS

Table of Contents

Chapter 04: Test and results 70
Introduction 74
1. Environment and developments tools 74
1.1. Google Colaboratory: 74
1.2. Data extraction: 74
1.2.1. Python 74
1.2.2. Json 75
1.2.3. Api client 75
1.3. Preprocessing 75
1.3.1. Pandas 75
1.3.2. NLTK 75
1.3.3. NumPy 75
1.3.4. Regular expression 76
1.3.5. Remove other language letters 76
1.4. Fine-tuning pretrained language models 76
1.4.1. Transformers Hugging Face: 76
1.4.2. Tensorflow 76
1.4.3. BertTokenizer 76
1.4.4. DistilBertTokenizer 77
1.4.5. Arabert Tokenizer 77
1.4.6. Bert for Sequence Classification 77
1.4.7. DistilBert for Sequence Classification 77
1.4.8. AraBERT for sequence classification 77
1.5. Training deep learning architecture 77
1.5.1. Keras 77
1.5.2. Genism 78
1.6. Performance Evaluation Metrics 78
2. Dataset composition 78
2.1. Our dataset 79
2.2. The second dataset 81
3. Test and evaluation 82
3.1. Training and Test Strategies 82
3.1.1. Hyper parameterization 82
3.1.2. Preprocessing for models 82
4. Test results for models 85

4.1. Machine learning models 85
4.1.1. With TF-IDF embedding 86
4.1.2. With Word2Vec embedding 87
4.2. Deep learning models 85
4.3. For transfer learning: 87
4.3.1. Bert layer with GRU and LSTM 92
4.4. Ensemble learning 95
4.5. Comparison between deep learning models and pretrained language models 97
4.6. General accuracy comparison 99
4.7. General f1_score comparison 100
Conclusion 103
Conclusion and perspective 104

Introduction

In this chapter, First, we describe the test environment and the various tools and libraries that were employed. then we go over the phases of our experimentation after illustrating the many steps we take to create a homogeneous dataset. Finally, we give the training model results and discuss them in order to evaluate our work.

1. Environment and developments tools

1.1. Google Colaboratory[15]:

Colaboratory, or "Colab" for short, is an interactive environment that allows writing and executing Python code on a browser, with no required configuration, free access to GPUs and TPU, and easy sharing. To get benefits of Google Colab, it is required to access a Gmail account in order to be able to create a Colab notebook where code can be written and executed. For machine and deep learning, Google colab provides all libraries and python modules needed to build, train and evaluate models. In addition, it is possible to install new modules by using the "pip install" command. Colab also enables loading data, and notebooks from local file systems, Google drive, and kaggle. Moreover, by using Google cloud storage services (. i.e., drive for example), there is a possibility to stock colab notebooks and other kinds of files. it also provides 12 hours of continuous execution for one session. Random Access Memory available space is 12 GB with no possibility of increasing freely, in addition to Google drive storage space, Google colab environment enables using over than 30 GB space for storage during a session, which means all python modules and files stored on this drive will be deleted when the session is over. It is possible to use CPU, GPU and TPU for execution. In our practical work, we chose to use GPUs for model training. Since they are very easy to use and they accelerate the execution of complex models.

1.2. Data extraction:

1.2.1. Python

We use python version 3.7.10 For collecting and scrapping the comments, Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high-level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. Python combines remarkable power with very clear syntax. It has interfaces to many systems calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface [16]. In our work we used as modules Json and API client discover

[15] Bienvenue dans Colaboratory - Colaboratory (google.com)
[16] Welcome to Python.org

1.2.2. Json

JSON stands for JavaScript Object Notation, is a lightweight format for storing and transporting data, it is often used when data is sent from a server to a web page JSON is "self-describing" and easy to understand. Python has a built-in package called json, which can be used to work with JSON data.

1.2.3. Api client[17]

To talk to a Google API, you need the Google APIs Client Library for Python, specifically the apiclient.discovery.build() function. In discovery.py is the build() function, which is what we need to create a service endpoint for interacting with an API.3.

1.3. Preprocessing

1.3.1. Pandas[18]

Pandas is an open source, easy-to-use data structures and data analysis tools for the Python programming language. This will help ensure the success of development of pandas as a world-class open-source project, and makes it possible to donate to the project. We used Pandas to read datasets csv/excel files and their data frames manipulation.

1.3.2. NLTK[19]

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum. We used the predefined NLTK list of stop words for Arabic.

1.3.3. NumPy[20]

is a python module distributed under a liberal BSD license which means that is a free module and its code is publicly available. NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more. Moreover, it has a high-level syntax and supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries. In our work, we use NumPy to define variables types, arrays reshaping, and models outputs stocking.

[17] API Cloud | Google Cloud | Google Cloud
[18] pandas - Python Data Analysis Library (pydata.org)
[19] Natural Language Toolkit — NLTK 3.6.2 documentation
[20] NumPy

1.3.4. Regular expression

Regular expressions (called REs, or regexes, or regex patterns) are essentially a tiny, highly specialized programming language embedded inside Python and made available through the re module[21]. We used it to remove punctuation, emoticon, URL, redundancy, words with less than two characters.

1.3.5. Remove other language letters

We used PyArabic (A specific Arabic language library for Python, provides basic functions to manipulate Arabic letters and text)[22] and alphabet-detector (A library to detect what alphabet something is written in)[23] to detecting Arabic letters, Arabic letters groups and characteristics, remove diacritics etc.

1.4. Fine-tuning pretrained language models

1.4.1. Transformers Hugging Face[24]:

known as pytorch transformers and pytorch-pretrained-bert. This module provides general-purpose architectures (BERT, GPT-2, RoBERTa, XLM, DistilBERT, XLNet…) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between TensorFlow 2.0 and PyTorch. This library is used to import pre-trained BERTbase model

1.4.2. Tensorflow

is an open-source library of software for dataflow and differential programming for various tasks. Similarly, TensorFlow is used in machine learning by neural networks. Developed by Google in 2011 under the name DistBelief, TensorFlow was officially released in 2017 for free. The library is able to run on multiple CPUs and GPUs and is available across multiple platforms. (Broughton et al., 2020).

1.4.3. BertTokenizer

The pretrained Bert model come with his own tokenizer BertTokenizer[25]. It is based on WordPiece tokenization, which makes it powerful at covering a wider spectrum of Out of Vocabulary (OOV) words. We use it on our approach during finetuning the pretrained Bert model with the multilingual parameter to get better performance on tokenizing our multilingual dataset.

[21] http://www.Python.org
[22] Welcome to PyArabic: Python Library for Arabic's documentation! — PyArabic: Python Library for Arabic 0.6.12 documentation
[23] alphabet-detector · PyPI
[24] Hugging Face – The AI community building the future.
[25] Tokenizer — transformers 4.10.0 documentation (huggingface.co)

1.4.4. DistilBertTokenizer

DistilBertTokenizer[26] comes with DistilBert pretrained model, works the same as BertTokenizer since it was not mentioned in the paper (Wolf et al.,2016) that the distillation was applied on the tokenizer part.

1.4.5. Arabert Tokenizer

Arabic tokenization comes with AraBert pretrained model chose WordPiece (Wu et al., 2016b) tokenizer as it was also used during the pre-training of BERT.

1.4.6. Bert for Sequence Classification

BertForSequenceClassification represents the Bert Model transformer with a sequence classification/regression head on top (a linear layer on top of the pooled output). This model is a PyTorch torch.nn.Module sub-class(Wolf et al., 2016). The model can be loaded and used from transformers library.

1.4.7. DistilBert for Sequence Classification

DistilBertForSequenceClassification, made available by the transformer's library as DistilBert model plus a classification layer on top. The model is loaded and used in the same way as the Bert model.

1.4.8. AraBERT for sequence classification

It takes the final hidden state of the first token, which corresponds to the word embedding of the special "[CLS]" token prepended to the start of each sentence. We then add a simple feed-forward layer with standard Softmax to get the probability distribution over the predicted output classes. During fine-tuning, the classifier and the pre-trained model weights are trained jointly to maximize the log-probability of the correct class.

1.5. Training deep learning architecture

1.5.1. Keras

Keras is a high-level neural networks library, written in Python and capable of running on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.[27]

[26] DistilBERT — transformers 4.10.0 documentation (huggingface.co)
[27] Developer guides (keras.io)

1.5.2. Genism[28]

Gensim is a free open-source Python library for representing documents as semantic vectors, it is designed to process raw, unstructured digital texts ("plain text") using unsupervised machine learning algorithms such as Word2Vec, FastText.

1.6. Performance Evaluation Metrics

Sklearn.metrics[29]: this scikit-learn module provides various metrics for the evaluation of machine and deep learning models performance. We choose to rely on this library in order to use functions that compute accuracy, recall, precision and f1 scores. In addition, we print matrix of confusion and classification reports by using this module.

2. Dataset composition

Since we want to study general hate speech in the online world, we collected and analyzed data from Facebook, Twitter and YouTube the most popular social media sites today. And we chose to collect data from different topics so we could test all the hate speech categories. In our work, we want to compare the performance of a multi-topic dataset and a unique topic dataset. For that reason, we are compelled to consume an available Algerian dialect dataset contain only one topic which is misogyny. Figure 36 explain the process

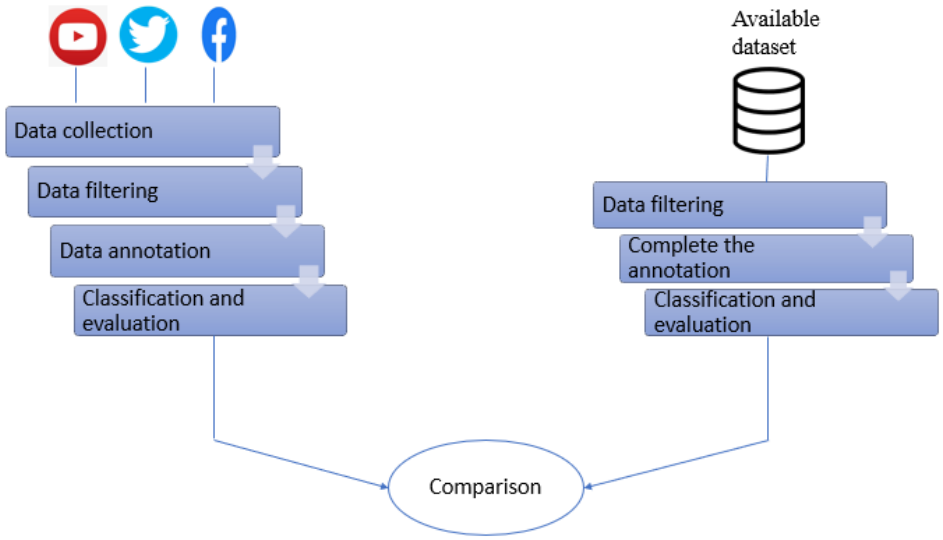


Figure 36: Process of comparing between the two datasets

[28] Gensim: Topic modelling for humans (radimrehurek.com)
[29] 3.3. Metrics and scoring: quantifying the quality of predictions — scikit-learn 0.24.2 documentation

2.1. Our dataset

Our dataset is made up of comments collected from various social media networks. We gathered 14150 comments from Facebook, Twitter, and YouTube, then filter and save them before manually annotating them to create the annotated dataset. Table 10. Information about the collected data.

Table 10: Information about the collected data.

YouTube	Facebook	Twitter
8357	5577	216

These comments cover a wide topics of hate speech and cyberbullying and abusive language, such as religion, parliamentary politics, Moroccan politics, sexism, and so on. We collect social media comments. Figure 37 shows the different topics.

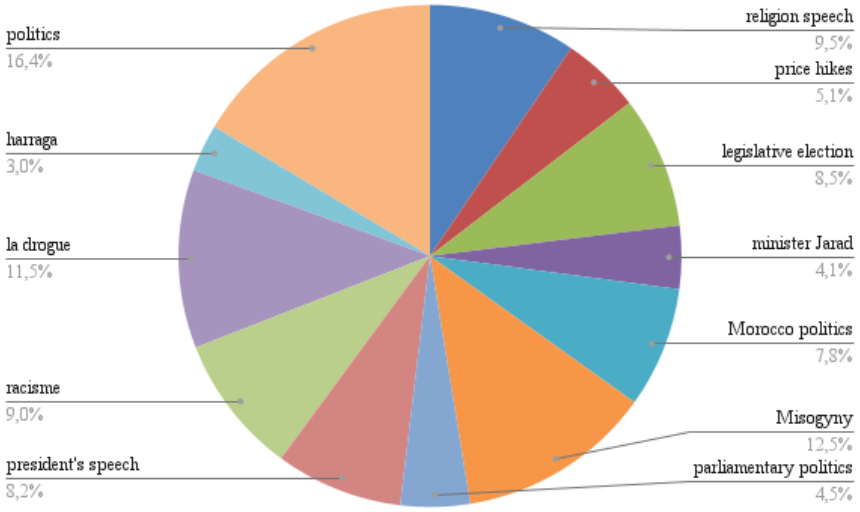
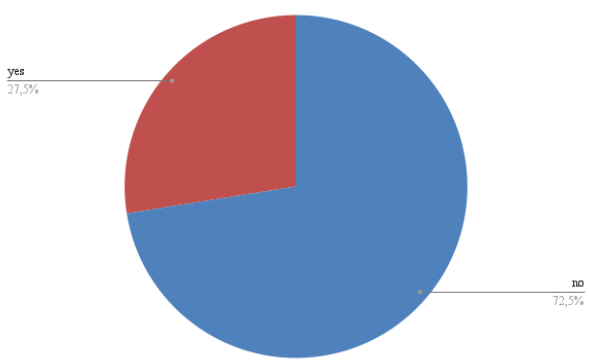


Figure 37: Dataset topics

The annotation was done by three annotators using three labels (hate speech, cyberbullying, abusive and offensive language), there are two classes for each label. yes for (hate/ bullying / abusive and offensive) and no for (non-hate/ non-bullying / non-abusive and offensive). Figure 38 shows data annotation, Figure 39,40,41 show the percentage of hate, cyberbullying and abusive language respectively.

Id	comments	Topic	hate speech	cyberbullying	nd Offensive	Source
id01	هههههه مهرج صوفي قبوري جاهل يفسر كما يحب واتباعه اجهل منه	religion speech	yes	no	yes	Youtube
id02	اسأل على اسمك يا مهرج العصر الرسول صلى الله عليه و سلم مقالناش ديروا عيد ميلادي لأنه من عمل اليهود والنصار	religion speech	yes	no	no	Youtube
id03	ياالحيواااان الدرس بدعة ياصوفي لماذا لاتتقى الله الله ياخذك اخذت عزيز مقتدر ياالجاهل	religion speech	yes	no	yes	Youtube
id04	الا فلتتق الله يا شمس الدين ؟ فان لم تفعل فليحذر الناس منك و من اقوالك ! و لا تتحدث عن السلفية هكذا فالكلمة ماخوذة من	religion speech	no	no	no	Youtube
id05	k Ya Cheikh Chemssedine oua Rebbi Ywafqak الله يهدينا جميع الى القران والسنة النبوية الشريفة امين	religion speech	no	no	no	Youtube
id06	ılblad nchri alkitab anta3k inchaalah manchtiwch anas almla7 alah yertha a3lik yachikh	religion speech	no	no	no	Youtube
id07	ياخي يهلول كان اجدر ان تضع انفا اصفر وتشفاه حمراء ولباس البهلوان ربي يهديك ولا يديك ويريح الأمة من جهلك وسفا	religion speech	yes	yes	yes	Youtube
id08	هذا تفكرت ديدين يكلاتشي لطفي اهيا المهرج ومراك داير قلي انيس وليد القومي	religion speech	yes	yes	no	Youtube
id09	دون سب أو شتم اقبلوا التحدي وتنوروورونا رعاكم الله حسبنا الله و نعم الوكيل	religion speech	no	no	no	Youtube
id10	اول قناة استحمارية هه من طمس الدين القبوري الجاهل للشيخ حمو الجاهل قناة الاستحمار	religion speech	yes	yes	yes	Youtube
id11	ياخسوف الكذابين والمنافقين والسارقين الفجرة الله افركس عظامك في حادث سير مروع انشاء الله الأهم انزل عليه مرض	religion speech	yes	no	yes	Youtube
id12	مليحة هاذي. لم يبق الا مهرج محدود العلم فاسد العقيدة ينطبق عليه القول دعاة على أبواب جهنم. لم يبقى الا مفتي قناة المهر	religion speech	yes	no	yes	Youtube
id13	عندما يتكلم شمس إسلام تتباح كلاب الوهبيون من كل بلدن رووح اتبيع للفت	religion speech	yes	no	yes	Youtube
id14	اكرهك في الله انت جدر من جدر البلاء بالله عليك اي احد من العلماء اتخذه العامة اضحوكة وصنفوه مع ممثلين كوميديين	religion speech	yes	yes	no	Youtube
id15	قرأت التعليقات مالقيت عيل السب و الهدرة الفارغة حتى واحد ماقال كلمة حق تقنع	religion speech	no	no	no	Youtube
id16	هذا بيتلكم خرابيبكم يا لوهابية اعداء الله يا رسول تكلم عليكم ب حديث و كل اعمالك و صفاتكم محلقين مقصرين	religion speech	yes	no	no	Youtube
id17	وين تلحق انت الشيخ فركوس..تع وجهك و نداع كمارتك..الشيخ تعلم في السعودية كي كانت قمجتك فرارة..لوكان سني يط	religion speech	yes	yes	no	Youtube
id18	المهرج شمس الدين منسيكتش مي نقلك ربي يهديك وشيخ الفركوس حفضه الله	religion speech	no	yes	yes	Youtube

Figure 38: Dataset annotation



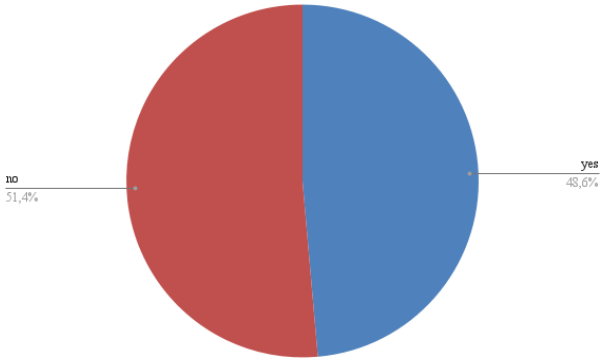


Figure 39: Percentage of cyberbullying

Figure 40: Percentage Hate speech

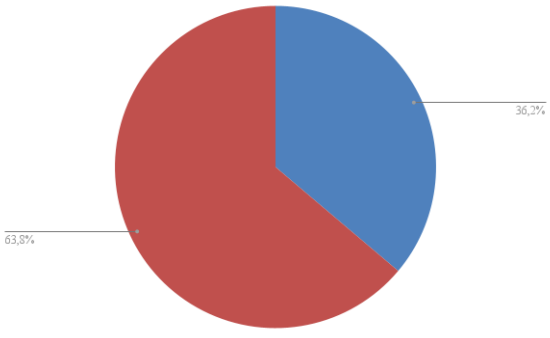


Figure 41: Percentage of abusive and offensive language

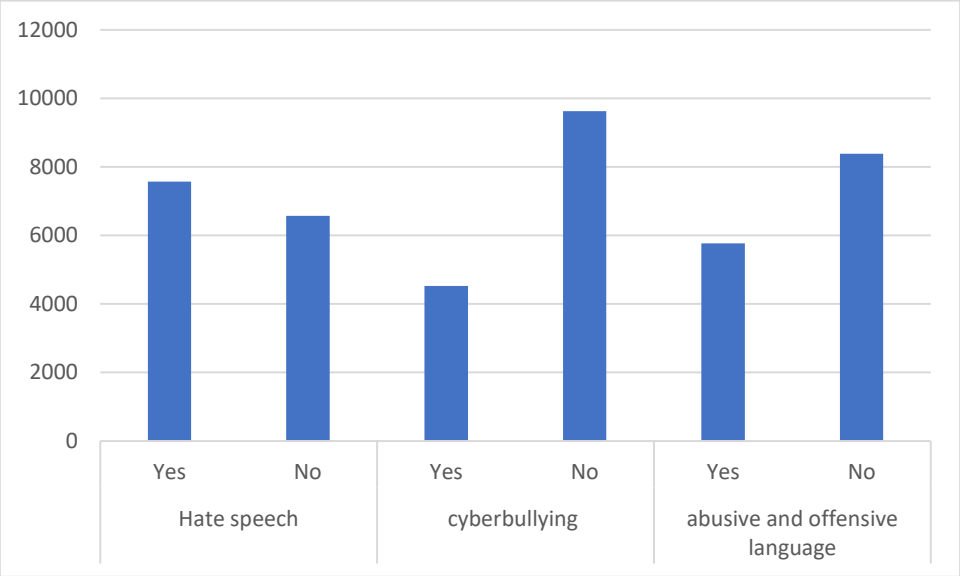


Figure 42: The dataset annotation

2.2. The second dataset

The misogyny dataset is a collection of 7396 labelled comments from Algerian social media posted online[30]. The dataset contains comments written in both Arabic and Latin characters. Its composition consists of three columns, one contains the comment (text), the other column represents the comment polarity. The comments are from one topic which is misogyny. The comment polarity varies between p and n, p for offensive and n for non-offensive. The dataset polarity distribution is shown in table 11.

Table 11: The dataset polarity

p	n
4301	3095

We delete all the repeated comments and we manually completed the annotation of this dataset for the sake of having multilabel dataset (hate speech, cyberbullying, abusive and offensive language).

[30] DjaziaH/Cyberbullying-Detection-DataSet: The first annotated dataset for the detection of cyberbullying and harassment in the Algerian online community. (github.com)

3. Test and evaluation

3.1. Training and Test Strategies

We opted to divide the dataset into 80 %, 10%, and 10% for training, validation, and test sets, respectively, by using stratified sampling to choose exactly 0.8, 0.1, and 0.1 sections of samples from the hate and non-hate classes.

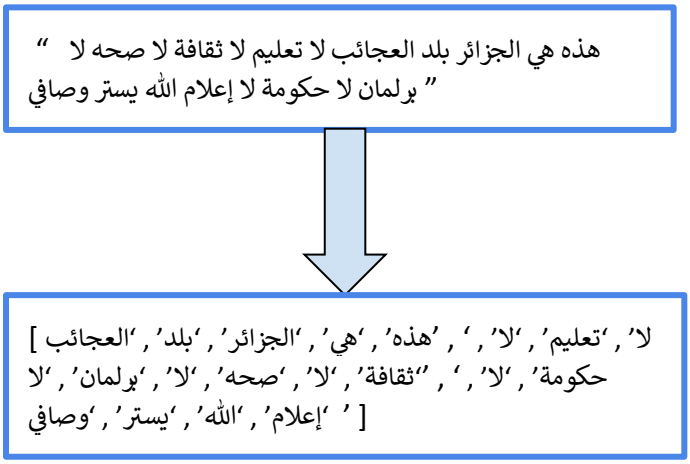
3.1.1. Hyper parameterization

To maintain the optimal version of the model, we must define appropriate hyper parameters such as batch size, learning rate, number of epochs, and optimizer: We choose 5 as the best number of epochs and 32 as the batch size for each epoch based on empirical data.

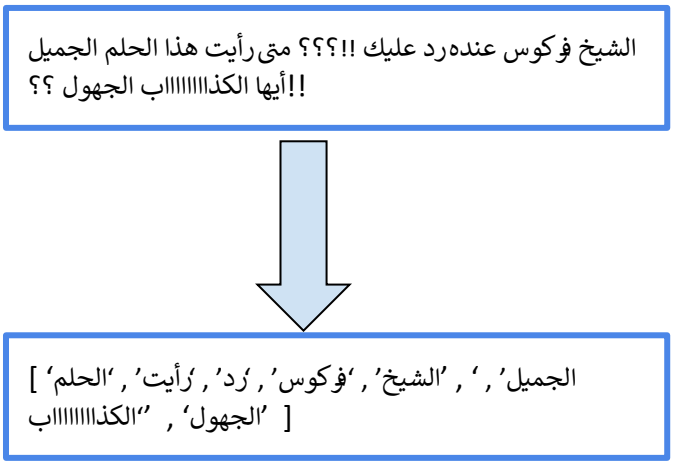
3.1.2. Preprocessing for models

For the preprocessing of comments before training models we started by keeping only the Arabic language using AlphabetDetector then removing all the emoticons using regular expressions, unifying letters in one specific form and removing the diacritics and the elongation and duplicated letters all using regular expressions and pyarabic library, then we filtered all the special characters such as hashtags, symbols, numbers, punctuation...etc. and removing stop words using nltk for the standard Arabic and creating Algerian stop words list finally deleting the empty lines. The graphics below provide an example of comment modification after each round of preprocessing. Following is an illustration of the preprocessing outcome of a provided comment example after each preprocessing step.

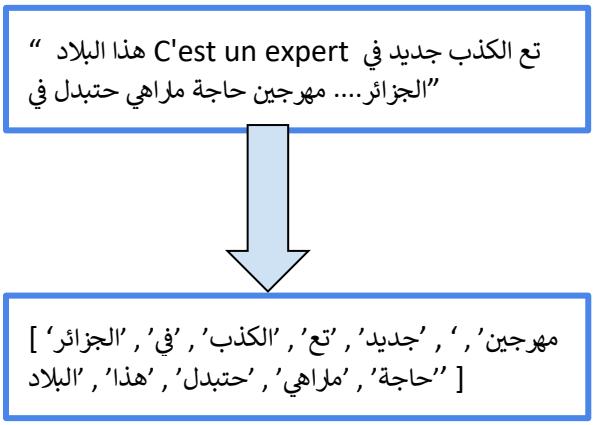
- **Tokenization**



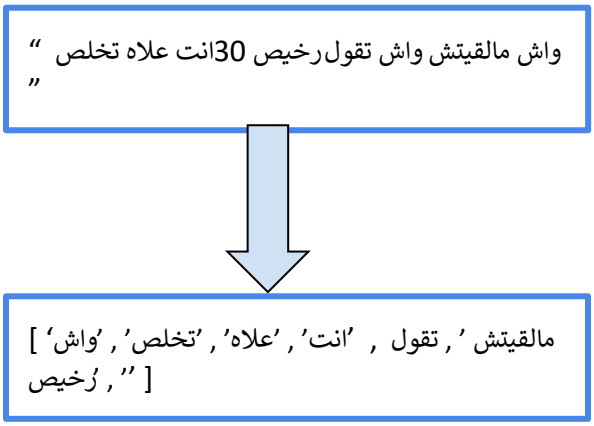
- **Removing stop words, Punctuations and special characters**



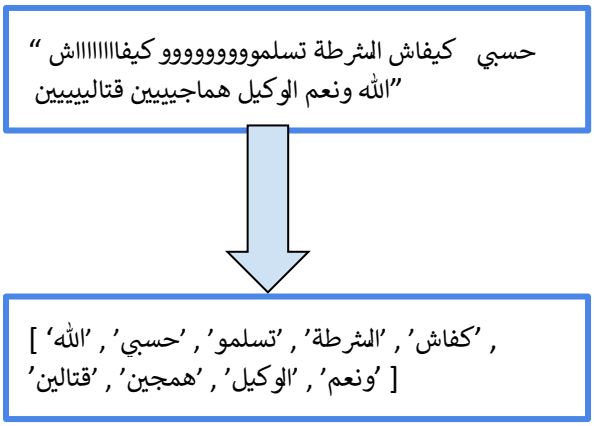
- **Remove other language letters**



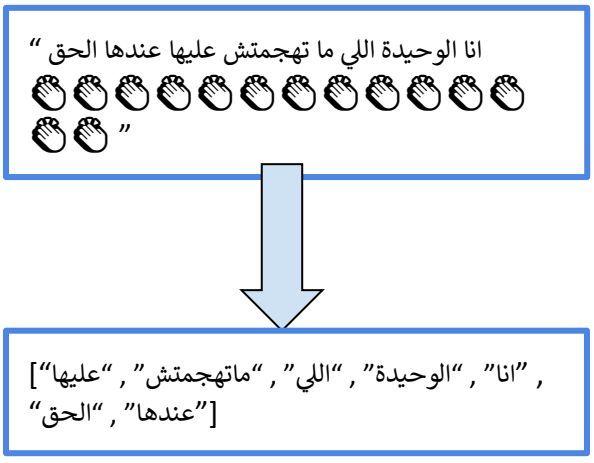
- **Removing numbers**



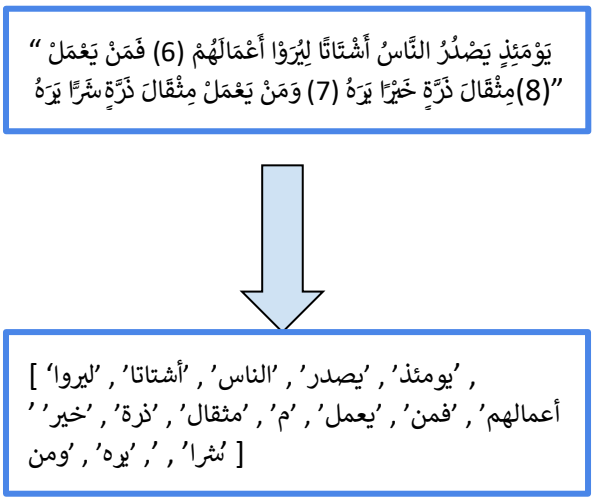
- **Removing duplicated letters**



- **Removing emoticons:**

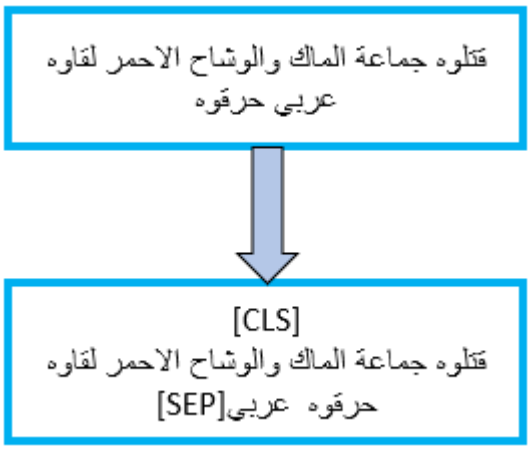


- **Removing diacritics:**



- **For transfer learning adding special tokens**

[SEP] and [CLS] special tokens are required to add in each sentence of the dataset, in order to express the beginning and the end of a sentence.



- **Input Ids and attention mask array**

After adding Bert's special tokens all the comments should be padded to max length, in our case, 40 tokens. then return the input ids and the attention mask as follow:

comments =تيزي وزو قاع كلابا و حاش لي مايستهلهاش

input ids =
[2,6018,2343,2146,1009,281,1771,21845,1006,251,22711,2099,2846,8429,13072,1042,3,0,0,0,,,]

attention mask = [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,,,,]

4. Test results for models

We run the different models on our dataset (MultiTopic-Dataset) before focusing on just one topic, misogyny dataset, which has over 6000 comments.

4.1. Machine learning models

We divided the dataset in two parts: test and training (test which represents 30% of dataset, and 70% training) to test the models and to calculate the accuracy. We use both TF-IDF and Word2Vec with five classification Algorithms such as: 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'). For their implementation phase, we were inspired by the classification algorithm proposed by (Joshi et

al., 2016). For the Word2vec model, we use the Gensim toolkit[31]. For FastText, we use the FastText library proposed by Facebook on Github6. For both Word2vec and FastText, we use a context of 50 words to produce representations for both CBOW and SG of length 100. we got the following test results:

4.1.1. With TF-IDF embedding

Table 13,14,15 show comparison of the test results between multiTopic dataset and misogyny dataset with the machine learning models once with TF-IDF and once with Word2vec (both CBOW and skip-gram) respectively.

Table 12: Results of TF-IDF with ML

Dataset	**Our Dataset**			**Misogyny Dataset**		
measure	**Accuracy**	**recall**	**Precision**	**Accuracy**	**Recall**	**Precision**
Randomforestclassifier	0.647	0.644	0.680	0.682	0.693	0.737
MultinomialNB	0.661	0.838	0.640	0.708	0.901	0.687
Logistic Regression	0.665	0.793	0.654	0.689	0.880	0.676
SGD Classifier	**0.670**	0.766	0.667	**0.711**	0.828	0.714
SVC classifier	0.666	0.727	0.674	0.704	0.849	0.700

[31] GenSim

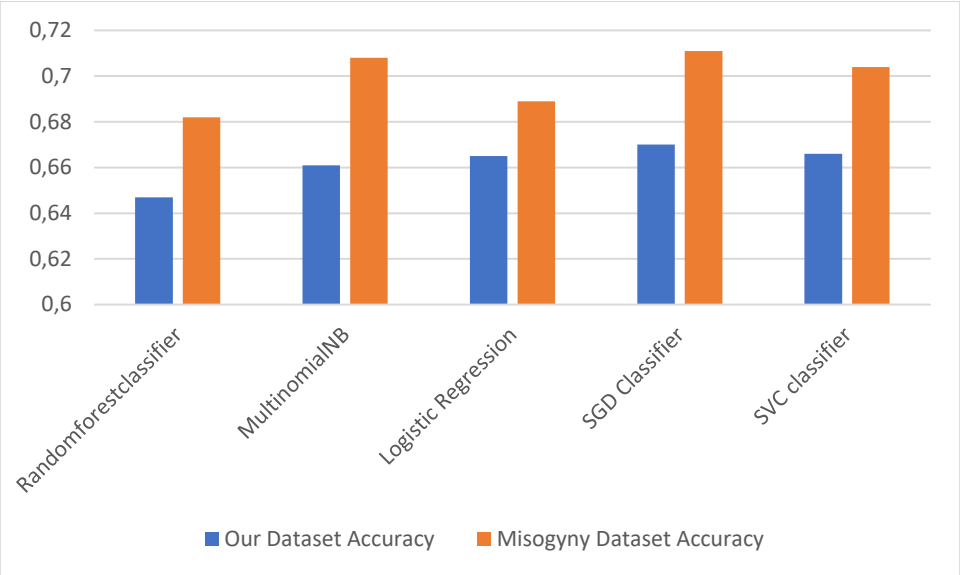


Figure 43: Comparison between multitopic and misogyny datasets for TF-IDF

4.1.2. With Word2Vec embedding

Table 13 shows the results of Skip-gram and SBOW with machine learning

Skip-Gram							SBOW					
Dataset	**Our Dataset**			**Misogyny Dataset**			**Our dataset**			**Misogyny dataset**		
measure	**Accuracy**	**recall**	**Precision**	**Accuracy**	**Recall**	**Precision**	**Accuracy**	**recall**	**Precision**	**Accuracy**	**Recall**	**Precision**
Random forestclassifier	0.555	0.611	0.579	**0.644**	0.766	0.665	0.561	0.624	0.584	**0.645**	0.768	0.665
MultinomialNB	0.534	0.935	0.537	0.580	0.928	0.584	0.551	0.707	0.564	0.616	0.846	0.622
Logistic Regression	0.556	0.757	0.563	0.601	0.919	0.599	0.558	0.702	0.570	0.622	0.868	0.622
SGD Classifier	0.543	0.444	0.599	0.613	0.867	0.615	0.560	0.936	0.552	0.610	0.909	0.607
SVC classifier	**0.557**	0.718	0.568	0.613	0.899	0.611	**0.563**	0.700	0.575	0.617	0.865	0.619

Table 13: the test results for SKIP-Gram and SBOW with ML

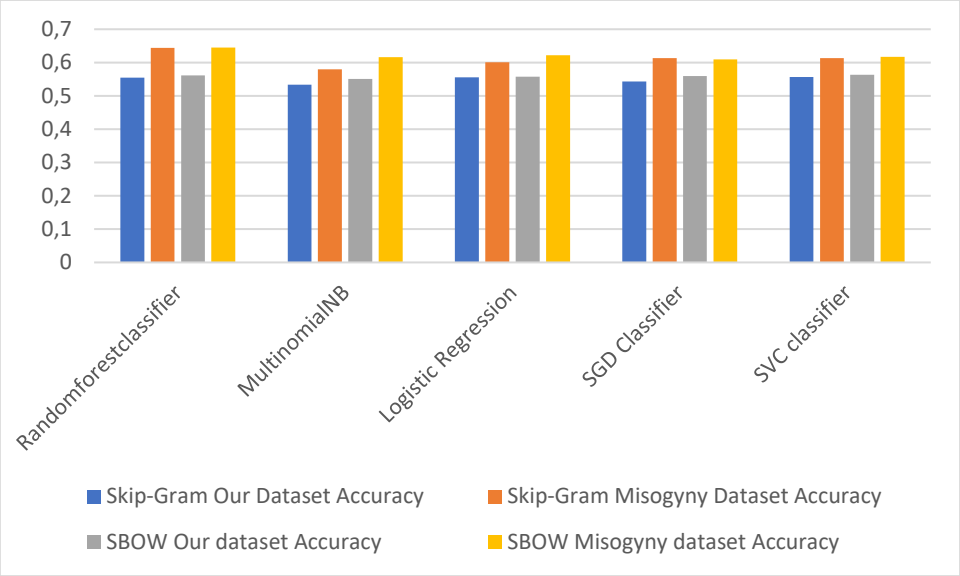


Figure 44: Comparison between multitopic and misogyny datasets for SBOW and Skip-Gram

Discussion

- We notice that TF-IDF embedding work better in each classifier with machine learning than Word2Vec embedding, the best accuracy with TF-IDF is 0.670 with our dataset and 0.711 with misogyny dataset while the best accuracy with W2V is 0.557 with our dataset and 0.563 with misogyny dataset.
- It is also remarkable that the Misogyny dataset has better performance than our dataset.

4.2. Deep learning models

For deep learning models we also split the dataset into 30% for testing and 70% for the training. Table 15 present the test results for multi-Topic dataset and misogyny dataset after testing the performers of the different deep learning models with FastText and without. For LSTM, GRU, Bi-LSTM or Bi-GRU, the first layer is a randomly-initialized word embedding layer that turns words in sentences into a feature map. The weights of the embedding matrix are calculated using FastText then the (LSTM/GRU/ Bi-LSTM / Bi-GRU) layer that scans the feature map. For reducing over-fitting a Dropout layer with a probability equal to 0.5 is added. The obtained scores are then feed to a single feed-forward (fully-connected) layer with Relu activation, the output of that layer goes through a sigmoid layer that predicts the output classes. We used Adam optimizers with epoch 10. For CNN, the first layer is a randomly-initialized word embedding layer that turns words in sentences into a feature map. The weights of the embedding matrix are calculated using FastText. then the CNN layer that scans the feature map. Global maxpooling is applied to the output layer to take the maximum score of each pattern. The obtained scores are then feed to a single feed-forward (fully-connected) layer with Relu activation, the output of that layer goes through a sigmoid layer that predicts the output classes. We used Adam optimizers with epoch 10

		Our Dataset					Misogyny dataset				
DL Models		**LSTM**	**GRU**	**Bi-LSTM**	**Bi-GRU**	**CNN**	**LSTM**	**GRU**	**Bi-LSTM**	**Bi-GRU**	**CNN**
Without FastText	**Accuracy**	**0.65**	**0.65**	0.64	0.63	0.52	0.70	0.70	0.70	**0.71**	0.57
	F1-Score	**0.71**	0.65	0.66	0.62	0.69	0.73	0.73	**0.74**	0.69	0.73
With FastText	**Accuracy**	0.65	0.65	**0.66**	**0.66**	0.54	**0.72**	0.71	0.70	0.71	0.58
	F1-Score	0.69	0.68	0.66	**0.71**	0.70	**0.75**	**0.75**	0.72	**0.75**	0.73

Table 15: Test results of the two datasets with and without FastTaxt

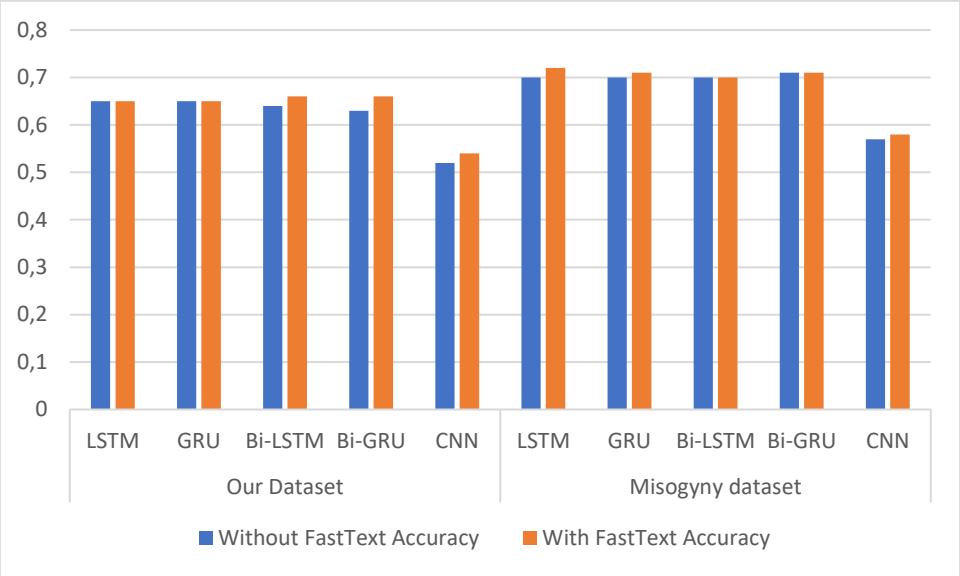


Figure 45: Comparison with FastText and without it for the two datasets

Discussion

- For deep learning models we can see that the RNN models perform better than CNN models.
- we can also see that LSTM and GRU perform better then Bi-LSTM and Bi-GRU in multi-topic and misogyny dataset.
- the best accuracy was 66% for bi-lstm and bi-gru with fasttext for our dataset, for misogyny dataset LSTM with FastText gives the best result with an accuracy of 71%.
- 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 chose to fine-tune Bert-base-arabic and Bert-base-multilingual-uncased and distilbert-base-uncased pre-trained models from Hugging Face transformers. Figure 46 presents the architecture of the Bert model. For each pre-trained model, we got the following test result in Table 16.

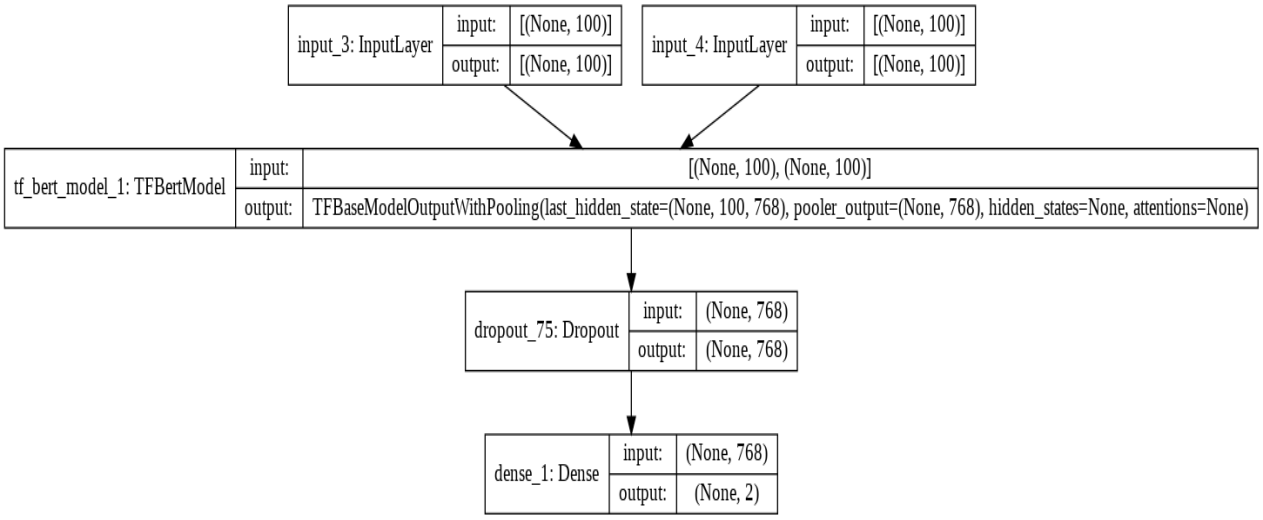


Figure 46: BERT architecture

Table 16: fine-tune results

Data	**our dataset**		**Misogyny dataset**	
Measure	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
Bert-base-arabic	**0.730**	**0.724**	**0.750**	**0.751**
Bert-base-multilingual-uncased	0.687	0.684	0.739	0.737
Distilbert-base-uncased	0.628	0.627	0.722	0.721

A. Misogyny dataset

The following figures show a comparison between training loss and validation loss for each model and a comparison between training and validation accuracy for the misogyny dataset

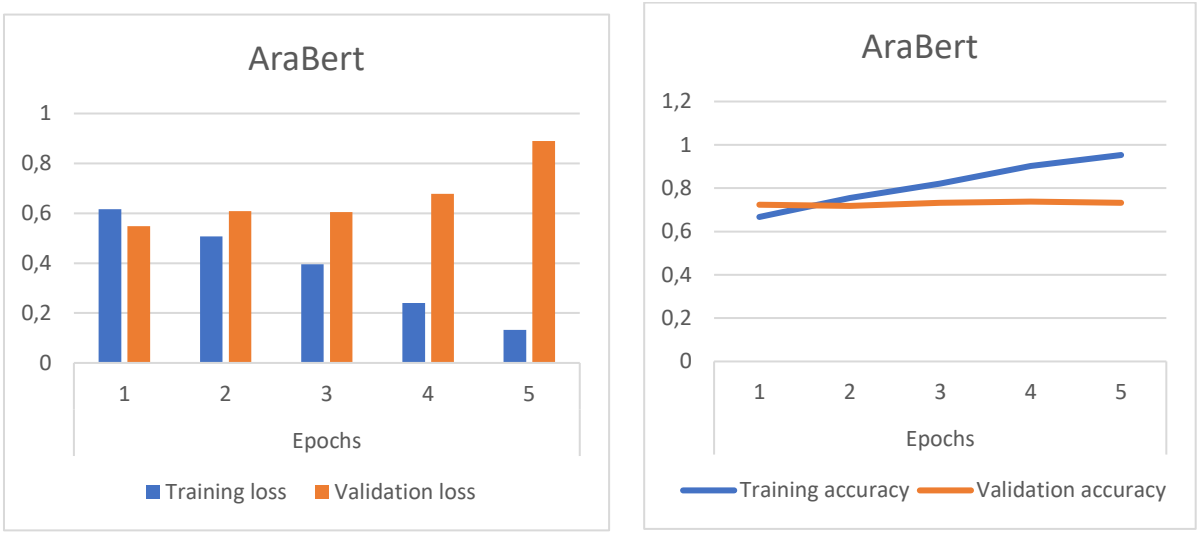


Figure 47: Arabert training and validation accuracy

Figure 48: Arabert loss and accuracy comparison.

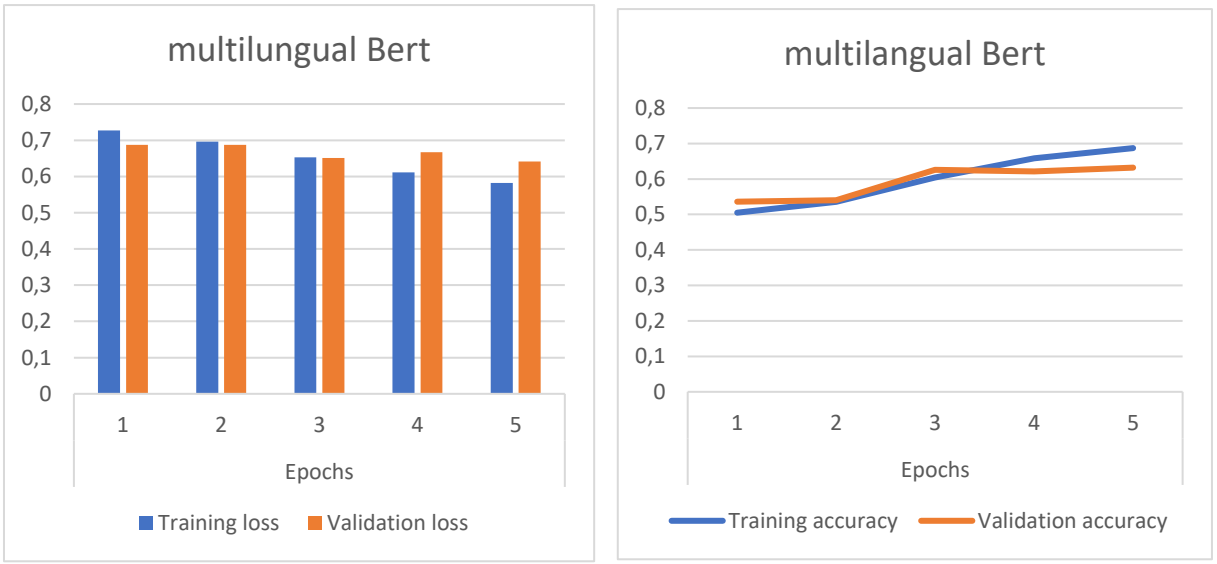
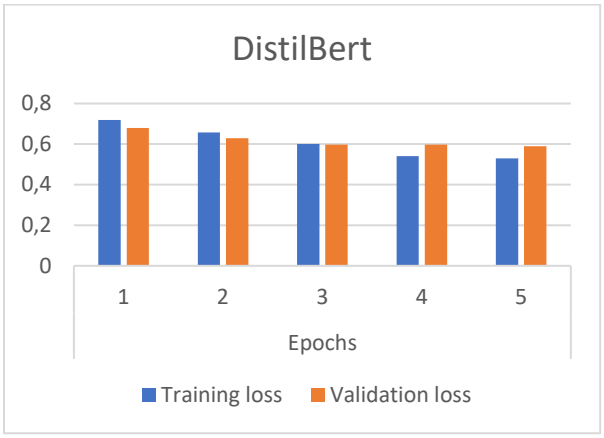


Figure 49: MBert training and validation accuracy.

Figure 50: MBert loss and accuracy comparison



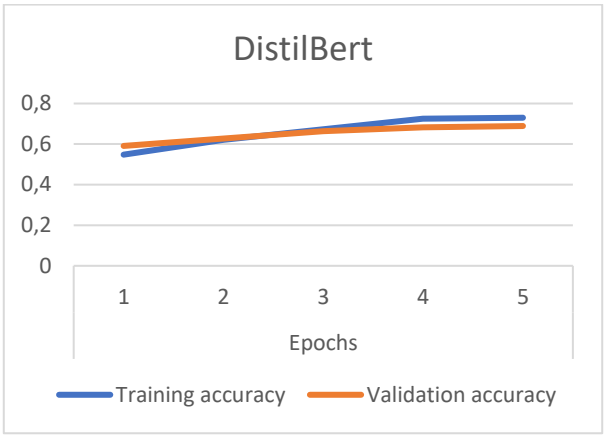
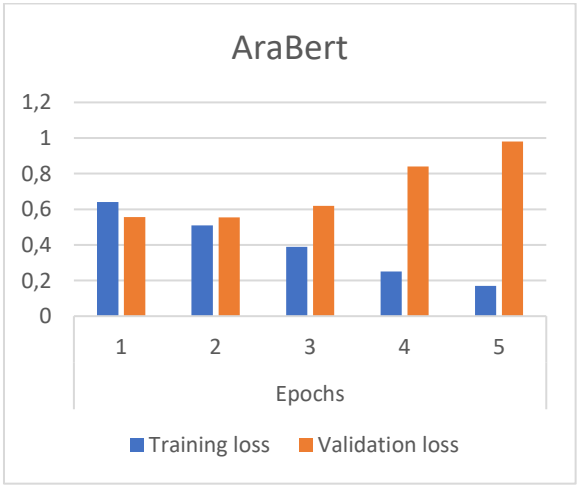


Figure 51: DistilBert loss and accuracy comparison

Figure 52: DistilBert training and validation accuracy.

B. Multitopic dataset

The following figures show a comparison between training and validation loss for each model and a comparison between training and validation accuracy for the multi-Topic dataset.



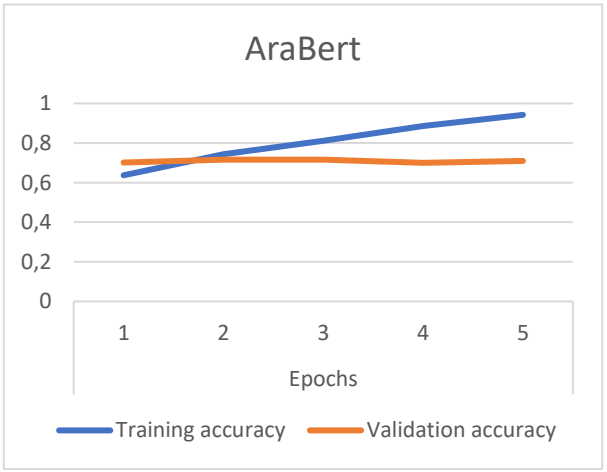
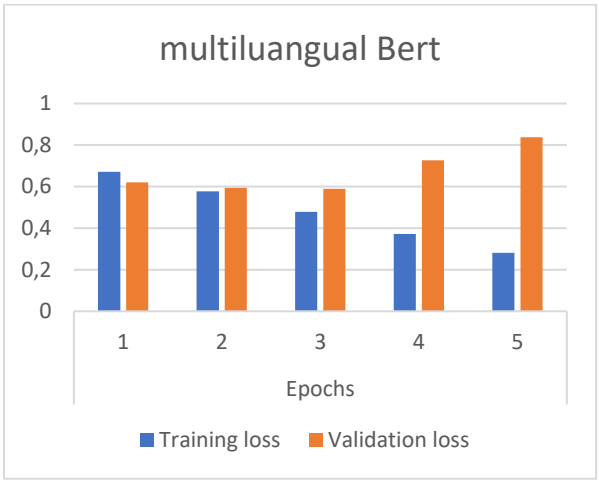


Figure 53: Arabert training and validation accuracy.

Figure 54: Arabert loss and accuracy comparison



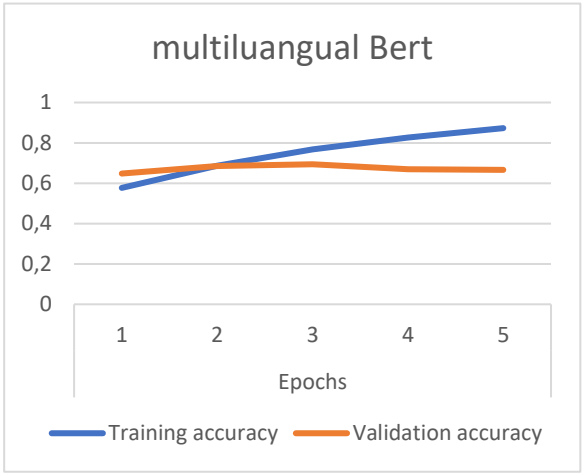
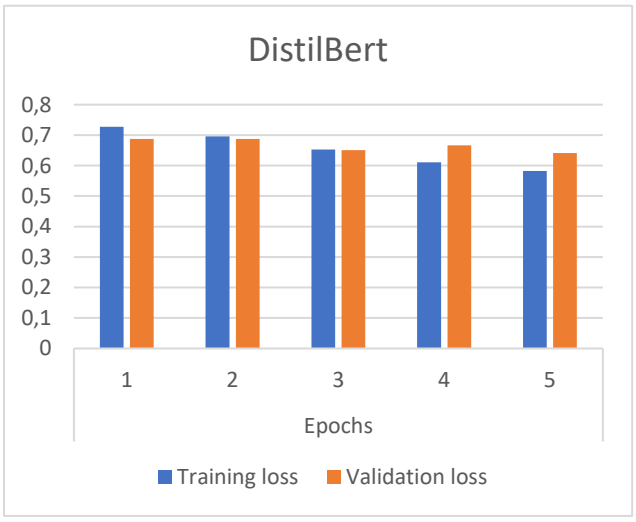


Figure 55: mBert training and validation accuracy.

Figure 56: mBert loss and accuracy comparison.



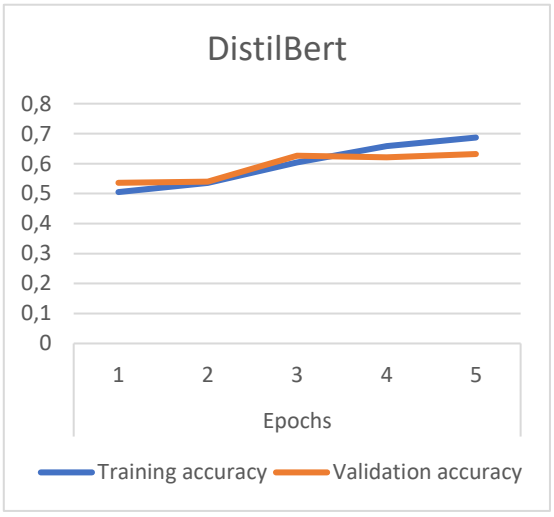


Figure 57: DistilBert training and validation accuracy comparison

Figure 58: DistilBert loss and accuracy

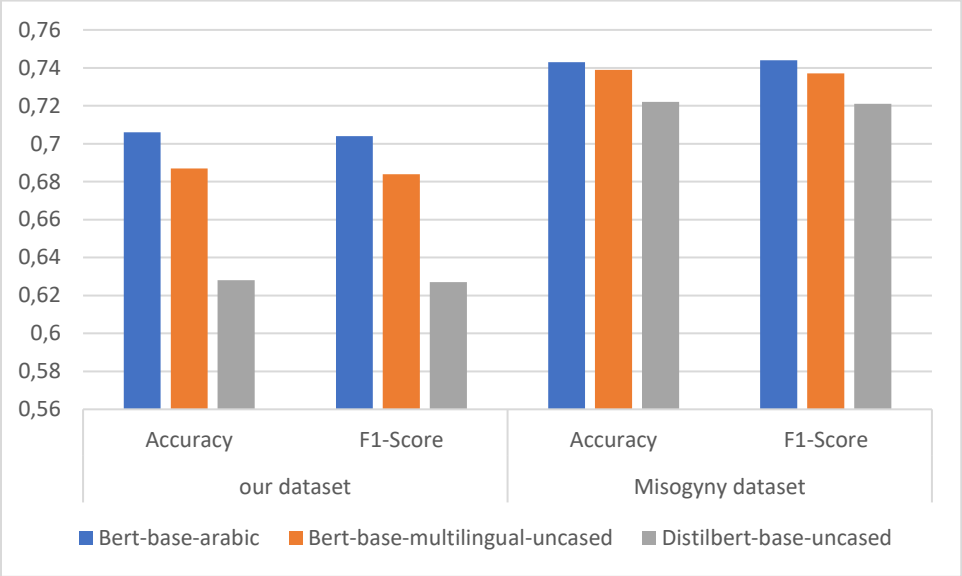


Figure 59: Misogyny and multitopic datasets accuracy and f1-score comparison for each model

Discussion:

- From Table 16 we see that the performance of pretrained language models on misogyny dataset varies from 72% to 75% and multitopic dataset varies from 62% to 73%.
- The Bert-base-Arabic model outperformed other models it achieved the best results with an accuracy of 70% and f1_score of 7% in our dataset.
- 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 the fine tune, we can use the BERT model as a layer to build a model with Tensorflow.keras library. In the following section we are going to present Bert with LSTM and Bert with GRU architectures.

We used all the hidden states produced by the recurrent layers (GRUs or LSTMs) in each time step. In a RNN layer, using all hidden states results for each token a concatenation of two vectors having the size of the number of units used in our case 192, where the first vector is for the forward RNN layer, and the second one is for the backward RNN layer. This is clearly presented in the figure 60, 61 and 62 where the outputs of the first and the second RNN layers having the size of (100, 384) for each sentence, are concatenated and then passed to a Dropout layer then to the final dense layer. Table 17 presents the results. the architecture of each model is presented in figure 60,61,62. The comparison of the two data is presented in figure 63.

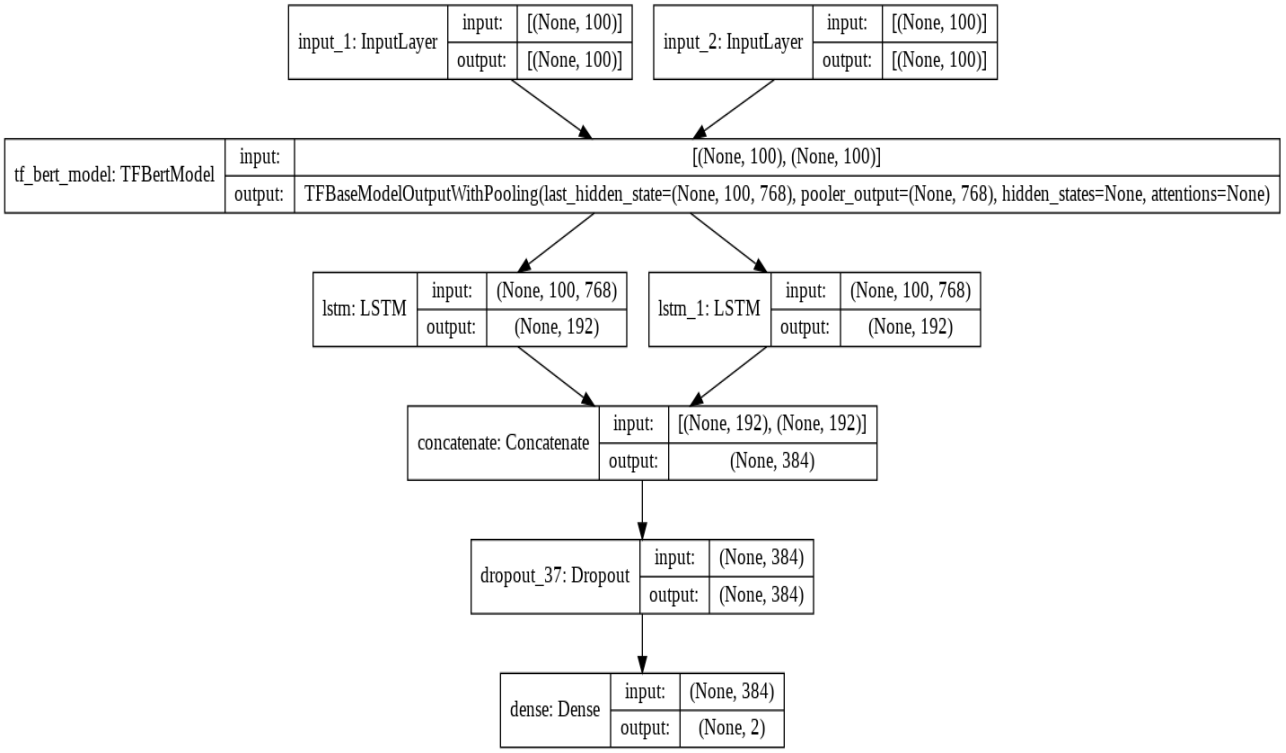


Figure 43: Bert with LSTM architecture

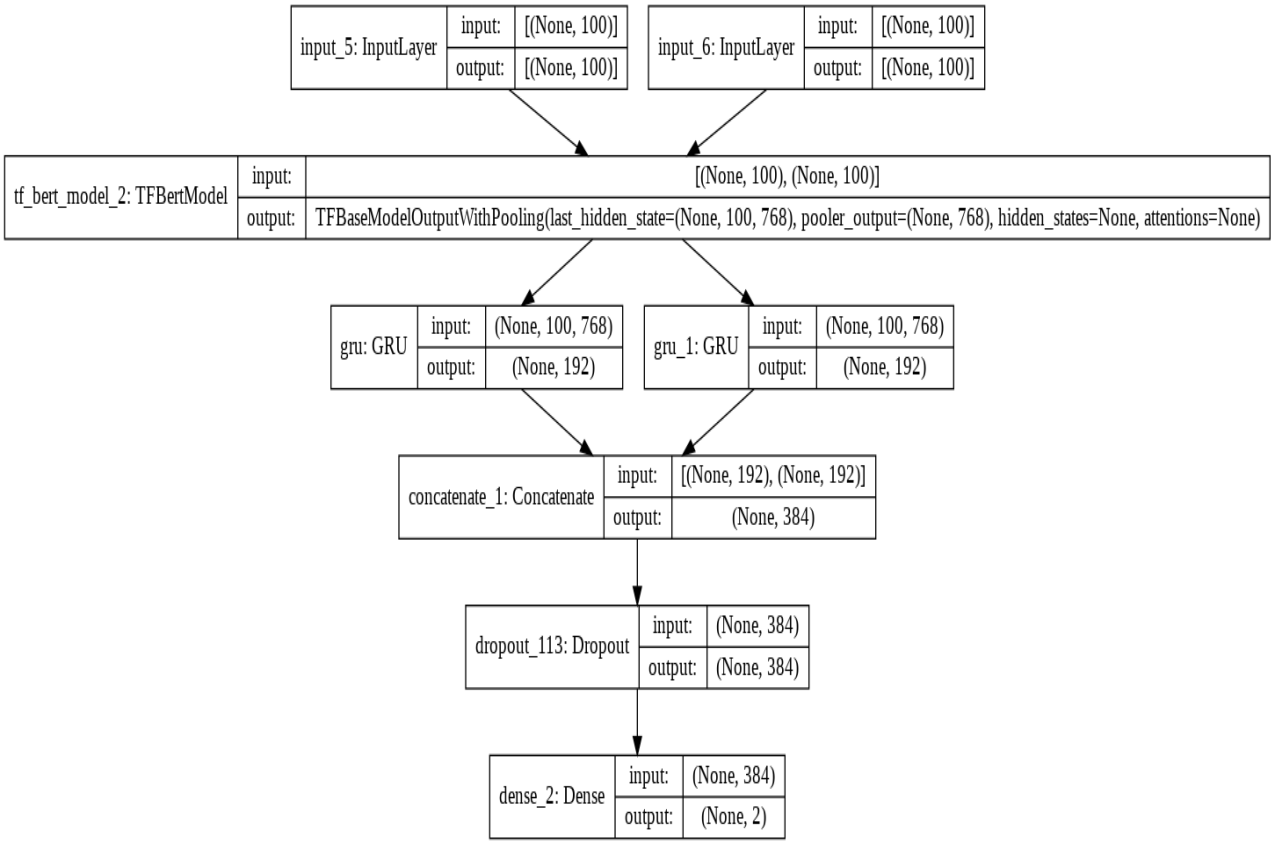


Figure 61: Bert with GRU architecture

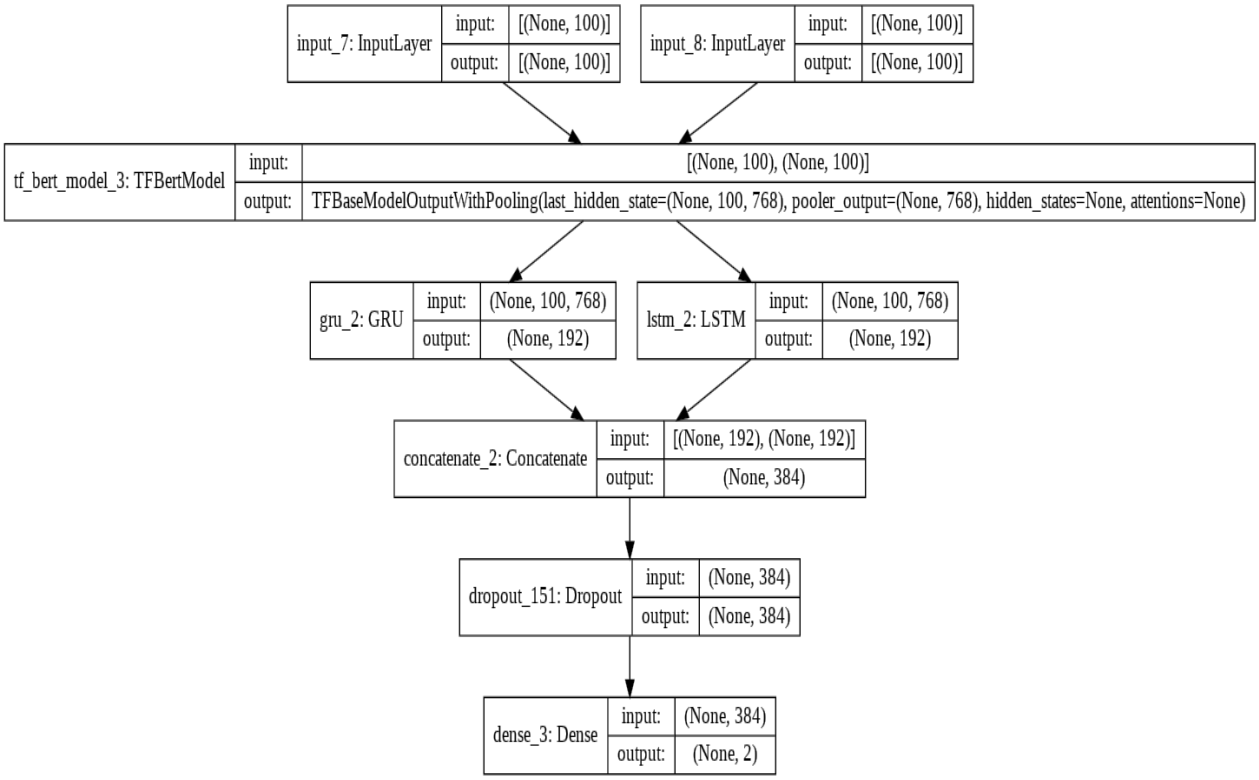


Figure 62: Bert with GRU and LSTM

Table 17: Bert embedding with GRU and LSTM results.

Dataset	**MultiTopic**		**Misogyny**	
Measure	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
BERT embedding +LSTM	0.72	0.72	0.73	0.72
BERT embedding +GRU	0.70	0.70	0.76	0.75
BERT embedding +GRU and LSTM	0.71	0.71	0.75	0.75

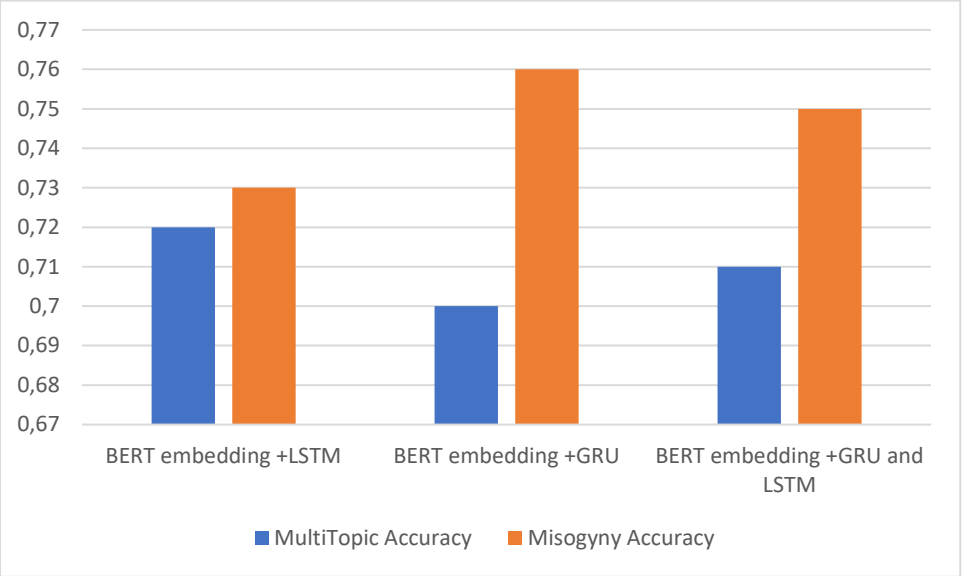


Figure 44: multitopic and misogyny datasets comparison between LSTM and GRU with Bert layer

Discuss

- We notice that adding Bert embedding gives better results with LSTM and GRU.
- The best accuracy was 72% for Bert embedding and LSTM for our dataset.
- For misogyny dataset GRU gives the best result with an accuracy of 76%.

4.4. Ensemble learning :

In this section, we propose building ensemble learning models composed of BERT model and other deep learning architectures based on LSTMs and GRUs. by combining BERT with LSTM, Bert with GRU and Bert with both LSTM and GRU.

In order to build our ensemble learning models, we train the models composing the ensemble model individually on the training set. Then, for classifying a new instance, we combine the outputs generated by each model of the ensemble by computing the mean for producing the ensemble learning model output. For the ordinary mean of outputs, we simply sum the ensemble models outputs and we divide the result by the number of these models.

Table 14: results of ensemble models.

Dataset	**MultiTopic**		**Misogyny**	
Measure	**Accuracy**	**F1-Score**	**Accuracy**	**F1-Score**
Model1: Bert+lstm	0.73	0.73	0.75	0.74
Model2 : Bert+ gru	0.72	0.72	0.75	0.74
Model3: Bert + both	0.72	0.72	0.76	0.76
Model1+Model2	0.73	0.73	0.76	0.76
Model1+Model3	0.73	0.72	0.76	0.76
Model2+Model3	0.72	0.72	0.75	0.75
Model1+Model2+model3	0.73	0.73	0.75	0.75
Bert + 3 models	**0.74**	**0.74**	**0.76**	**0.76**

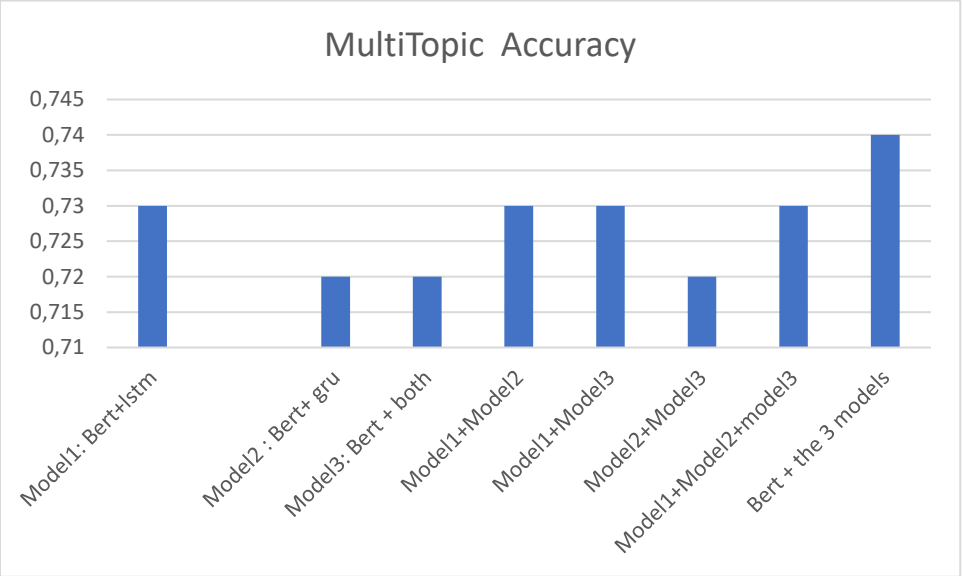


Figure 45: multitopic and misogyny dataset accuracy, f1-score comparison for ensemble learning

Discussion

- For ensemble learning we can see that ensemble model 1 perform better then model 2 and model 3 in both datasets with an accuracy of 73% and f1_score of 73%.
- The misogyny dataset gives the best result with an accuracy of 76% and f1_score of 76%.
- We can see that our final ensemble models have improved the best results with 74% accuracy and 74% f1-score
- We notice that ensemble models have improved the accuracy and f1_score in both datasets.

4.5. Comparison between deep learning models and pretrained language models

Table 15 shows the different between the deep learning models and the pretrained language models.

Table 15: deep learning / pretrained language models Comparison

Dataset	**Our dataset**	
	Accuracy	**F1-Score**
LSTM	0.65	0.71
GRU	0.65	0.65
Bi-LSTM	0.64	0.64
Bi-GRU	0.63	0.62
CNN	0.52	0.69
Bert-base-arabic	**0.73**	**0.72**
Bert-base-multilingual-uncased	0.68	0.68
Distilbert-base-uncased	0.62	0.62

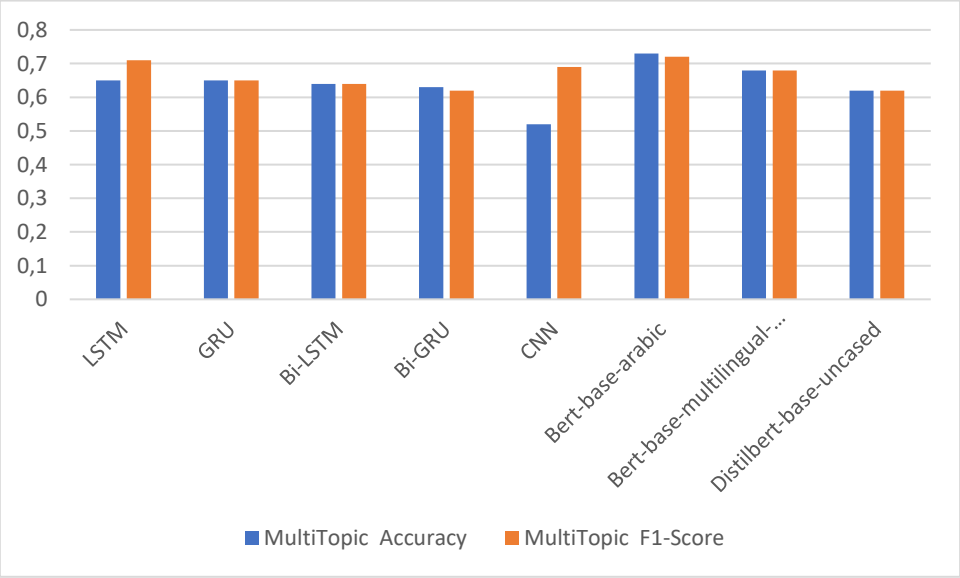


Figure 46: deep learning / pretrained language models Comparison

Discussion

- From the previous table we can see that the pretrained models performed much better than the deep learning models.
- We notice that bert-base-arabic achieve the best accuracy in both datasets.

4.6. General accuracy comparison

Table 16: General accuracy comparison between all the models

Dataset	MultiTopic	Misogyny	Percentage
LSTM	0.65	0.70	7.14%
GRU	0.65	0.70	7.14%
Bi-LSTM	0.64	0.70	8.57%
Bi-GRU	0.63	0.71	11.27%
CNN	0.52	0.57	8.77%
Bert-base-arabic	0.73	0.75	2.67%
Bert-base-multilingual-uncased	0.68	0.73	6.85%
Distilbert-base-uncased	0.62	0.72	13.89%
BERTembedding+ LSTM	0.72	0.73	1.37%
BERTembedding+ GRU	0.70	0.76	7.89%
Model1: bert+lstm	0.73	0.75	2.67%
Model2 : bert+ gru	0.72	0.75	4%
Model3: bert + both	0.72	0.76	5.26%
Model1+	0.73	0.76	3.95%

Model2			
Model1+ Model3	0.73	0.76	3.95%
Model2+ Model3	0.72	0.75	4%
Model1+ Model2+ model3	0.73	0.75	2.67%
Bert + 3 models	**0.74**	**0.76**	2.63%

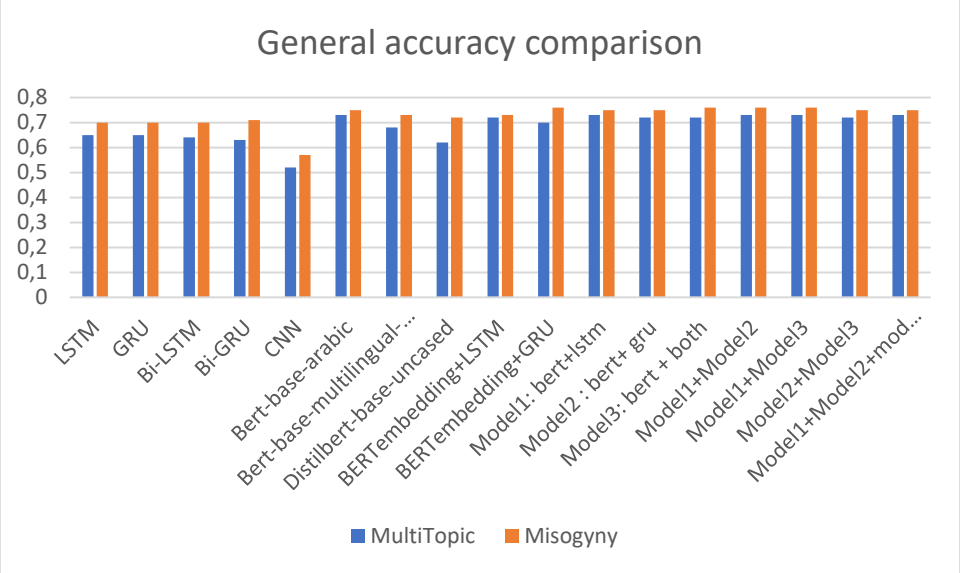


Figure 47: General accuracy comparison

4.7. General f1_score comparison

Table 17: General f1_score comparison between all the models

Dataset	MultiTopic	Misogyny	percentage
LSTM	0.71	0.70	1.43%
GRU	0.65	0.70	7.14%
Bi-LSTM	0.66	0.70	5.71%
Bi-GRU	0.62	0.71	12.68%

CNN	0.69	0.57	21.05%
Bert-base-arabic	0.72	0.75	4%
Bert-base-multilingual-uncased	0.68	0.73	6.85%
Distilbert-base-uncased	0.62	0.72	13.89%
BERTembedding+LSTM	0.72	0.72	0%
BERTembedding+GRU	0.70	0.75	6.67%
Model1: Bert+lstm	0.73	0.74	1.35%
Model2 : Bert+ gru	0.72	0.74	2.70%
Model3: Bert + both	0.72	0.76	5.26%
Model1+Model2	0.73	0.76	3.95%
Model1+Model3	0.72	0.76	5.26%
Model2+Model3	0.72	0.75	4%
Model1+Model2+model3	0.73	0.75	2.67%
Bert + 3 models	**0.74**	**0.76**	2.63%

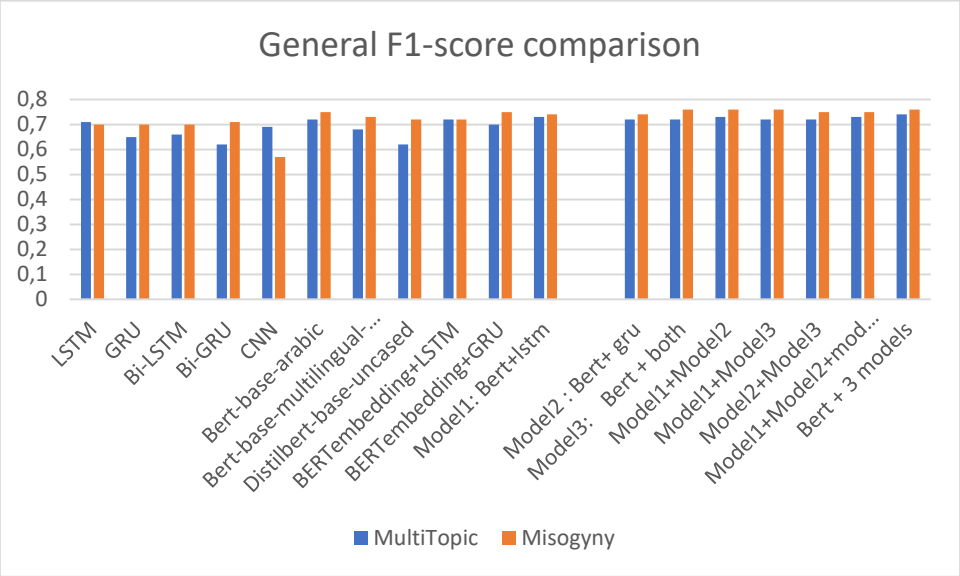


Figure 48 : General f1_score comparison

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 67% 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 71% accuracy and 70% F1-Score.
- we can also see that LSTM and GRU perform better then Bi-LSTM and Bi-GRU in multi-topic and misogyny dataset.
- the best accuracy was 66% for bi-lstm and bi-gru with fasttext for our dataset, for misogyny dataset LSTM with FastText gives the best result with an accuracy of 71%.
- 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.
- The Bert-base-Arabic model achieved the best results with an accuracy of 73% and f1_score of 72% in our dataset.
- We notice that the performance of pretrained language models achieved highly better results on both datasets.
- It is also remarkable that the Misogyny dataset has better performance than our dataset.
- Another conclusion is that merging various high-performing classifiers that have been trained individually improves the prediction capacity of the resulting ensemble learning classifier.
- For ensemble learning we can see that ensemble model 1 perform better then model 2 and model 3 in both datasets with an accuracy of 73% and f1_score of 73%.
- The misogyny dataset gives the best result with an accuracy of 76% and f1_score of 76%.
- We can see that our final ensemble models have improved the best results with 74% accuracy and 74% f1-score
- We notice that ensemble models have improved the accuracy and f1_score in both datasets.

Conclusion

In this chapter, we've discussed tests and evaluations of our proposed approaches, as well as the various training models and data collecting modules that have been utilized. The best version of our trained models from scratch earned an accuracy and f1_score of 65% and 71%, respectively in our tests.

Conclusion and perspective

Conclusion

Conclusion and perspective

Our work makes a great effort to try to fight against the spread of hate speech in the Algerian dialect on the social media. We presented an approach for dealing with the problems in that language. first, we create a dataset by collecting 14150 comments from social media: Facebook, YouTube and Twitter (7573 comments are positive and 6577 are negative) for hate speech and (4525 comments are positive and 9625 are negative) for cyberbullying and (5768 are positive and 8384 are negative) for offensive and abusive language. These comments are from different topics such as politic, misogyny, racism and many others.

Preprocessing is the next stage. Where we have studied the characteristics of the Arabic language in general and the Algerian dialect in particular and its complexity, we proposed to filter the non-Arabic comments, tokenization, removing stop words, removing duplicated letters, and removing punctuation and special characters…etc. then to improve our dataset we test it on some of the machine learning algorithms once with TF-IDF and once with word2vec embedding layer, the best result we reached with machine learning was with the SGD classifier with 0.67 accuracy for the multitopic dataset and 0.71 accuracy with the misogyny dataset , we also test it on the deep learning models and the best result was with Bi-LSTM and Bi-GRU with FastText embedding with 0.66 for the multiTopics dataset. Then we chose to fine-tune Bert-base-arabic model and we got 0.73 accuracy for the MultiTopic dataset and 0.75 f1-score for the Misogyny dataset as a best result. Finally, we proposed our ensemble learning model composing of 3 ensemble learning models the first one is BERT model with Bert embedding and LSTM the second one is BERT with Bert embedding and GRU the third one is BERT with Bert embedding and LSTM with GRU. We got 0.74 accuracy and f1-score for the MultiTopics dataset and 0.76 accuracy and f1-score for the Misogyny dataset.

The different avenues explored during this work have led us to consider many perspectives. We present here those which seem the most important to us. We will try to generalize this work on the other Algerian dialects such as (Tamazigth, Chaoui, Chelhi, Targui…). Also, we will try to increase the size of the dataset in where each topic contains 10k comments as minimum size. Also avoiding comments with the same meaning in the collecting phase, and we will try to use other pretrained models with other robust architecture and last and not least we will try to create a pre trained model for the Algeria dialect.

Bibliography

Abozinadah, E. A., Mbaziira, A. v., & Jones, J. H. Jr. (2015). Detection of Abusive Accounts with Arabic Tweets. *International Journal of Knowledge Engineering-IACSIT*, *1*(2), 113–119. https://doi.org/10.7763/IJKE.2015.V1.19

Agarwal, S., & Sureka, A. (2015). *Using KNN and SVM Based One-Class Classifier for Detecting Online Radicalization on Twitter BT - Distributed Computing and Internet Technology* (R. Natarajan, G. Barua, & M. R. Patra, Eds.; pp. 431–442). Springer International Publishing.

Albadi, N., Kurdi, M., & Mishra, S. (2018a). *Are They Our Brothers ? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere*.

Albadi, N., Kurdi, M., & Mishra, S. (2018b). *Are They Our Brothers ? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere*.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, *8*(3), 1–67. https://doi.org/10.3390/electronics8030292

Alsafari, S., Sadaoui, S., & Mouhoub, M. (2020). Hate and offensive speech detection on Arabic social media. *Online Social Networks and Media*, *19*(March), 100096. https://doi.org/10.1016/j.osnem.2020.100096

Alwakid, G., Osman, T., & Hughes-Roberts, T. (2017). Challenges in Sentiment Analysis for Arabic Social Networks. *Procedia Computer Science*, *117*, 89–100. https://doi.org/10.1016/j.procs.2017.10.097

Blaya, C. (2018). *Version of Record: https://www.sciencedirect.com/science/article/pii/S1359178918300028*. 1–35.

Broughton, M., Verdon, G., McCourt, T., Martinez, A. J., Yoo, J. H., Isakov, S. v., Massey, P., Halavati, R., Niu, M. Y., Zlokapa, A., Peters, E., Lockwood, O., Skolik, A., Jerbi, S., Dunjko, V., Leib, M., Streif, M., von Dollen, D., Chen, H., … Mohseni, M. (2020). *TensorFlow Quantum: A Software Framework for Quantum Machine Learning*. http://arxiv.org/abs/2003.02989

Chen, Y., Zhou, Y., Zhu, S., & Xu, H. (2012). Detecting offensive language in social media to protect adolescent online safety. *Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust and 2012 ASE/IEEE International Conference on Social Computing, SocialCom/PASSAT 2012*, 71–80. https://doi.org/10.1109/SocialCom-PASSAT.2012.55

Chetty, N., & Alathur, S. (2018a). Hate speech review in the context of online social networks. *Aggression and Violent Behavior*, *40*(March 2017), 108–118. https://doi.org/10.1016/j.avb.2018.05.003

Chetty, N., & Alathur, S. (2018b). Hate speech review in the context of online social networks. *Aggression and Violent Behavior*, *40*(March 2017), 108–118. https://doi.org/10.1016/j.avb.2018.05.003

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019a). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(Mlm), 4171–4186.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019b). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(Mlm), 4171–4186.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. http://arxiv.org/abs/1810.04805

Faris, H., Aljarah, I., Habib, M., & Castillo, P. A. (2020). Hate speech detection using word embedding and deep learning in the Arabic language context. *ICPRAM 2020 - Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods*, *March*, 453–460.

https://doi.org/10.5220/0008954004530460
Fortuna, P., & Nunes, S. (2018). A survey on automatic detection of hate speech in text. In *ACM Computing Surveys* (Vol. 51, Issue 4). Association for Computing Machinery. https://doi.org/10.1145/3232676
Freund, L. E., & Shah, K. M. (1994). A Survey of Simulation Courses in Industrial Engineering Programmes. *European Journal of Engineering Education*, *19*(1), 93–103. https://doi.org/10.1080/03043799408923274
Gregorie, T. M. (2001). Cyberstalking: Dangers on the information superhighway. *National Center for Victims of Crime*, *1*(2), 1–9.
Guellil, I., Adeel, A., Azouaou, F., Benali, F., Hachani, ala-eddine, & Hussain, A. (2018). *Arabizi sentiment analysis based on transliteration and automatic corpus annotation*. 335–341. https://doi.org/10.18653/v1/P17
Guellil, I., Adeel, A., Azouaou, F., Boubred, M., Houichi, Y., & Abdelhaq Moumna, A. (2021). Sexism detection: The first corpus in Algerian dialect with a code-switching in Arabic/ French and English. *Arxiv.Org*. http://www.esi.dz
Hewitt, S., Tiropanis, T., & Bokhove, C. (2016). The problem of identifying misogynist language on Twitter (and other online social spaces). *Proceedings of the 8th ACM Conference on Web Science*.
Hosseini, H., Kannan, S., Zhang, B., & Poovendran, R. (2017). Deceiving Google's perspective API built for detecting toxic comments. *ArXiv*.
Huang, Q., Zhang, L., Cheng, Y., Li, P., & Li, W. (2018). Enantioselective Construction of Vicinal Sulfur-containing Tetrasubstituted Stereocenters via Organocatalyzed Mannich-Type Addition of Rhodanines to Isatin Imines. *Advanced Synthesis and Catalysis*, *360*(17), 3266–3270. https://doi.org/10.1002/adsc.201800642
ILGA. (2016a). *Europe annual review*.
ILGA. (2016b). *Europe annual review*.
Joshi, J. B. D., Institute of Electrical and Electronics Engineers, & IEEE Computer Society. (2016). *Proceedings, 2016 IEEE International Conference on Big Data : Dec 05-Dec 08, 2015, Washington D.C., USA*.
Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). *Bag of Tricks for Efficient Text Classification*. http://arxiv.org/abs/1607.01759
Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2017). Bag of tricks for efficient text classification. *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, *2*(2), 427–431. https://doi.org/10.18653/v1/e17-2068
Jourová, V. (2017). *Speech by Commissioner Jourová - 10 years of the EU Fundamental Rights Agency: a call to action in defence of fundamental rights, democracy and the rule of law*. *February*.
Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, *8*(3), 1–67. https://doi.org/10.3390/electronics8030292
Alwakid, G., Osman, T., & Hughes-Roberts, T. (2017). Challenges in Sentiment Analysis for Arabic Social Networks. *Procedia Computer Science*, *117*, 89–100. https://doi.org/10.1016/j.procs.2017.10.097
Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 1412–1421. https://doi.org/10.18653/v1/d15-1166

Mathew, B., Illendula, A., Saha, P., Sarkar, S., Goyal, P., & Mukherjee, A. (2020). Hate begets Hate: A Temporal Study of Hate Speech. *Proceedings of the ACM on Human-Computer Interaction*, *4*(CSCW2). https://doi.org/10.1145/3415163
McNamee, L. G., Peterson, B. L., & Peña, J. (2010a). A call to educate, participate, invoke and indict: Understanding the communication of online hate groups. *Communication Monographs*, *77*(2), 257–280. https://doi.org/10.1080/03637751003758227

McNamee, L. G., Peterson, B. L., & Peña, J. (2010b). A call to educate, participate, invoke and indict: Understanding the communication of online hate groups. *Communication Monographs*, *77*(2), 257–280. https://doi.org/10.1080/03637751003758227

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. http://arxiv.org/abs/1301.3781

Mozafari Marzieh and Farahbakhsh, R. and C. N. (2020). A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media. In S. and M. J. F. and M. E. and R. L. M. Cherifi Hocine and Gaito (Ed.), *Complex Networks and Their Applications VIII* (pp. 928–940). Springer International Publishing.

Mubarak, H., Darwish, K., & Magdy, W. (2017). *Abusive Language Detection on Arabic Social Media*. 52–56. https://doi.org/10.18653/v1/w17-3008

Mubarak, H., Darwish, K., Magdy, W., Elsayed, T., & Al-Khalifa, H. (2020). Overview of {OSACT}4 {A}rabic Offensive Language Detection Shared Task. *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, *May*, 48–52.

Mulki, H., Haddad, H., Bechikh Ali, C., & Alshabani, H. (2019). *L-HSAB: A Levantine Twitter Dataset for Hate Speech and Abusive Language*. 111–118. https://doi.org/10.18653/v1/w19-3512

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016a). Abusive language detection in online user content. *25th International World Wide Web Conference, WWW 2016*, 145–153. https://doi.org/10.1145/2872427.2883062

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016b). Abusive language detection in online user content. *25th International World Wide Web Conference, WWW 2016*, 145–153. https://doi.org/10.1145/2872427.2883062

Park, J. H., & Fung, P. (2017). *One-step and Two-step Classification for Abusive Language Detection on Twitter*. 41–45. https://doi.org/10.18653/v1/w17-3006

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018a). Deep contextualized word representations. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*, 2227–2237. https://doi.org/10.18653/v1/n18-1202

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018b). Deep contextualized word representations. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*, 2227–2237. https://doi.org/10.18653/v1/n18-1202

Raisi, E., & Huang, B. (2016a). *Cyberbullying Identification Using Participant-Vocabulary Consistency*. 46–50.

Raisi, E., & Huang, B. (2016b). *Cyberbullying Identification Using Participant-Vocabulary Consistency*. 46–50.

Ross, B., Rist, M., Carbonell, G., Cabrera, B., Kurowsky, N., & Wojatzki, M. (2017). Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. *ArXiv*. https://doi.org/10.17185/duepublico/42132

Schmidt, A., & Wiegand, M. (2017). *A Survey on Hate Speech Detection using Natural Language Processing*. *2012*, 1–10. https://doi.org/10.18653/v1/w17-1101

Sohn, H., & Lee, H. (2019). MC-BERT4HATE: Hate speech detection using multi-channel bert for different languages and translations. *IEEE International Conference on Data Mining Workshops, ICDMW*, *2019-November*, 551–559. https://doi.org/10.1109/ICDMW.2019.00084

Tarasova, N. (2016). Classification of Hate Tweets and Their Reasons using SVM. *Uppsala University*.

Van Hee, C., Lefever, E., Verhoeven, B., Mennes, J., Desmet, B., De Pauw, G., Daelemans, W., & Hoste, V. (2015). Detection and fine-grained classification of cyberbullying events. *International Conference Recent Advances in Natural Language Processing, RANLP*, *2015-Janua*, 672–680.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 5999–6009.

Williams, M. L., & Burnap, P. (2016). Cyberhate on Social Media in the aftermath of Woolwich: A Case Study in Computational Criminology and Big Data. *British Journal of Criminology*, *56*(2), 211–238. https://doi.org/10.1093/bjc/azv059

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., le Scao, T., Gugger, S., … Rush, A. M. (2020). *Transformers: State-of-the-Art Natural Language Processing*. https://github.com/huggingface/

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., … Dean, J. (2016a). *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 1–23.

Wu, Y., Schuster, M., Chen, Z., Le, Q. v., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., … Dean, J. (2016b). *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. http://arxiv.org/abs/1609.08144

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019a). Predicting the type and target of offensive posts in social media. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(July), 1415–1420. https://doi.org/10.18653/v1/n19-1144

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019b). Predicting the type and target of offensive posts in social media. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(July), 1415–1420. https://doi.org/10.18653/v1/n19-1144