
Instructions for Mac Users

ARABIC OFFENSIVE LANGUAGE DETECTION IN SOCIAL MEDIA

by

Fatemah Ali Husain

A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

_____ Dr. Ozlem Uzuner, Dissertation Director

_____ Dr. Syed Abbas Zaidi, Committee Member

_____ Dr. Amarda Shehu, Committee Member

_____ Dr. Hadi El-Amine, Committee Member

_____ Dr. Deborah Goodings, Associate Dean

_____ Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date:_____ Spring Semester 2021
George Mason University
Fairfax, VA

Arabic Offensive Language Detection in Social Media

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Fatemah Ali Husain
Master of Science
College of Science and Engineering, University of Minnesota, 2015
Master of Science
College of Graduate Studies, Kuwait University, 2009

Director: Ozlem Uzuner, Professor
Department of Information Technology

Spring Semester 2021
George Mason University
Fairfax, VA

DEDICATION

"لَولا بُنيات كَزَغب القَطا -- حَطَطنَ مِن بَعضٍ إلى بَعضِ
لكان لي مُضطَرَبٌ واسِعٌ -- في الأرضِ ذاتِ الطولِ والعرضِ
وإنَّما أولادُنا بَينَنا -- أكبادُنا تَمشي علَى الأرضِ
إنْ هَبَّتْ الرِّيحُ على بَعضِهم -- لَمْ تَشبعْ العينُ مِن الغَمضِ"
الشاعر: حطان بن المعلى

إلى علي، و محمد، و ليلى ...

ACKNOWLEDGEMENTS

In the name of Allah (God), the most beneficent, the most merciful

Prophet Muhammad (peace be upon him and his holy progeny) said: “The pen of a writer is mightier than the blood of a martyr.”

The true reward lies with Allah, but I would like to sincerely thank Kuwait University for funding my study and staff at Kuwait University and Kuwait Cultural Office, who provided generous support and help during my struggles with oppressors and haters at George Mason University. I never truly knew hate before coming to George Mason University. However, while the university taught me a lesson about extremism in hate, Islam taught me to overcome hate peacefully with patience and silence while working hard to let my actions speak louder than my words.

I would like to thank my dissertation committee members, Dr.Uzuner, Dr.Shehu, Dr.Zaidi, and Dr.El-Amine. This dissertation would not have been possible without their guidance, help, and support.

I am very thankful to my mentor at Google Research, Dr.Alyssa Lees, and my colleagues who provide great support during my study, including Angelica, May, Noha, Mansour, Boshra, Pattiya, Hui, Chola, Arman, Yaya, Jamie, Paul, Giridhar, and Krystal.

شكرا أمي الحبيبة...

شكرا أبي العزيز...

شكرا زوجي و أولادي و بنتي الأعزاء...

شكرا لكل أسرتي وصديقاتي الغاليين...

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS AND SYMBOLS

Abusive Score ..... AbS
Adjective ..... A
Adverbial Phrase ..... AP
African American English ..... AAE
Area Under Receiver Operating Characteristic ..... AUROC
Artificial Intelligence ..... AI
Artificial Neural Network ..... ANN
Association for Computational Linguistics ..... ACL
Association for Computing Machinery Digital Library ..... ACM DL
Bag-Of-Means ..... BOM
Bag-of-Words ..... BOW
Bayes Networks ..... BN
Bi-Normal Separation ..... BNS
Bidirectional Encoder Representations from Transformers ..... BERT
Bidirectional Gated Recurrent Unit ..... Bi-GRU
Bidirectional LSTM ..... Bi-LSTM
Bidirectional RNN ..... Bi-RNN
Chi-Square CHI2
Classical Arabic Language ..... CAL
Co-occurrence MAP ..... CM-SPAM
Complement NB ..... CNB
Conjunction C
Contemporary Corpus of Arabic ..... CCA
Continuous Bag-Of-Word ..... CBOW
Convolutional Neural Network ..... CNN
Convolutional Neural Network - Bidirectional Long Short-Term Memory. CNN-BiLSTM
CrowdFlower ..... CF
Discriminative Multinomial Naïve Bayes ..... DMNB
Equivalence class based sequential Rule Miner ..... ERMiner
Exploratory Data Analysis ..... EDA
False Negatives ..... FN
False Positives ..... FP
Feed Forward Neural Network ..... FFNN
First Word ..... x^{ι}
Gated Recurrent Unit ..... GRU
Gaussian Process Latent Variable Model ..... GPLVM

General Data Protection Regulations in Europe ..... GDPR
Global Vector ..... GloVe
Hate Score ..... HtS
Information Gain ..... IG
Information Retrieval ..... IR
Institute of Electrical and Electronics Engineers ..... IEEE
Inverse Document Frequency ..... IDF
Islamic State of Iraq and Syria ..... ISIS
k-Nearest Neighbors ..... kNN
Language Resources and Evaluation Conference ..... LREC
Latent Dirichlet Allocation ..... LDA
Leave-One-Platform-Dataset-Out ..... LOPO
Levantine Twitter Dataset for Hate Speech and Abusive Language ..... L-HSAB
Log Odds Ratio ..... LOR
Logistic Regression ..... LR
Long Short Term Memory Network ..... LSTM
Machine Learning ..... ML
Masked Language Model ..... MLM
Mean ..... μ
Modern Standard Arabic ..... MSA
Multi-labeled Reuters-21578 Dataset R8
Multi-Platform Offensive Language Dataset ..... MPOLD
Multilingual BERT ..... M-BERT
Multinomial Naive Bayes ..... MNB
Naive Bayes ..... NB
Named Entity Recognition ..... NER
Natural Language Processing ..... NLP
Next Sentence Prediction ..... NSP
Noun ..... N
Noun Phrase ..... NP
Offensive Language Identification Datasets ..... OLID
Open Source Arabic Corpus ..... OSAC
Open Source International Arabic News Corpus ..... OSIAN
Open-Source Arabic Corpora and Corpora Processing Tools ..... OSACT
Out-Of-Word-Embeddings-Vocabulary ..... OOWEV
PageRank ..... PR
Parametric Rectified Linear Unit ..... PReLU
Part-of-Speech ..... POS
Pearson's chi-square test ..... x^2
Pointwise Mutual Information ..... PMI
Preposition ..... P
Prepositional Noun Phrase ..... PNP
Principle Component Analysis ..... PCA
Rectified Linear Unit ..... ReLU

Recurrent Neural Network ..... RNN
Sample Mean ..... $\bar{x}$
Sample Size ..... N
Sample Variance ..... s^2
Second Word ..... y^{ι}
Semantic Orientation ..... SO
Stanford Natural Language Inference Corpus ..... SNLI
Stochastic Gradient Descent ..... SGD
State-Of-The-Art ..... SOTA
Support Vector Machine ..... SVM
TensorFlow Research Cloud ..... TFRC
Term Frequency ..... TF
Term Frequency-Inverse Document Frequency ..... TF-IDF
Top-K Sequential pattern mining ..... TKS
Tree-Structured Parzen Estimator ..... TPE
True Negatives ..... TN
True Positives ..... TP
Tunisian Hate and Abusive Speech Dataset ..... T-HSAB
Universities Data Set ..... WebKB
Variance ..... σ^2
Variant of SVM ..... SMO
Verb V

ABSTRACT

ARABIC OFFENSIVE LANGUAGE DETECTION IN SOCIAL MEDIA

Fatemah Ali Husain, Ph.D.

George Mason University, 2021

Dissertation Director: Dr. Ozlem Uzuner

Studies have shown that cyberhate, online harassment, and use of offensive language in social media are on the rise. Use of offensive language, even online, creates an exclusive environment and can even foster real-world violence. Despite significant technological advances in Natural Language Processing (NLP), online offensive language detection remains one of the most challenging text classification tasks due to the ambiguity and informality of the language used in user-generated content, as well as the social context of the users. Automatic offensive language detection is further complicated in languages with diverse forms and limited resources, such as Arabic. Arabic is spoken widely in the Middle East, encompasses multiple dialects and cultures, and represents a multitude of nationalities. The wide-adoption and the heterogeneity of Arabic affects also the social context that allows the interpretation and automatic recognition of offensive language.

This dissertation proposes and develops methods for automatic offensive language detection for Arabic in social media. It explores transfer learning approaches to tackle this challenging task in a resource-constrained language that is rich in dialectal and cultural variations. Our studies show that: (1) advanced language models can capture the ambiguous, informal variations in the user generated text and accurately recognize offensive language; (2) different Arabic dialects can inform each other and significantly contribute to cross-dialect offensive language detection; (3) different user-generated content platforms do not add value to offensive language detection across platforms; (4) we can customize existing state of the art language models to improve their coverage of dialectal Arabic; and (5) dialectal Arabic language model outperforms the non-dialectal model on some offensive language detection datasets.

INTRODUCTION

> "وَقُولُوا لِلنَّاسِ حُسْنًا"
> "And you shall speak to men good words."
>
> -Holy Qur'an, chapter Al-Baqara (2), verse 83

The increasing number of online platforms for user-generated content enable more people to experience the freedom of expression than ever before. In addition, users of these platforms can be anonymous and hide their identity, which can increase the chance of misusing these platforms. Online offensive language creates an exclusive environment and in more severe cases, it can foster real-world violence (Sap et al., 2019). The use of offensive language has become one of the most common problems on social networking platforms. According to a study by the Pew Research Center in 2015, 67% of people in the US agree they should be able to publicly make offensive statements against minority groups (Laub, 2019). Some countries have issued laws to ban hate speech, a kind of offensive language, on social networking platforms. For example, in 2017, Germany passed the Network Enforcement Act, a law that requires social media companies to remove hate speech from their websites (Kent, 2018). In addition to legislative reforms, technological solutions have been adopted to enforcing these reforms.

Wiedemann et al. (2018) describe offensive language as "threats and discrimination against people, swear words or blunt insults" (p.1). Whether a statement is offensive can also depend on its context. For example, while the word 'oriental' in the phrase 'oriental food' is acceptable, the same word in 'oriental man' is not. Accordingly, it is important to consider the context and the intention of the writer when defining offensive language. Social networking platforms exhibit multiple offensive language types, including hate speech, aggressive content, cyberbullying, and toxic comments (Mironczuk & Protasiewicz, 2018).

Natural Language Processing (NLP) techniques, such as text classification, can automatically detect offensive language. Text classification refers to the process of assigning pre-defined labels to new text (Mironczuk & Protasiewicz, 2018).

Machine Learning (ML), Artificial Intelligence (AI), Deep Learning (DL), and NLP techniques provide ample opportunities to develop a model capable of automatically detecting offensive language online. The primary purpose of this dissertation is to provide a design for an NLP system that can utilize the State-Of-The-Art (SOTA) and build on it for Arabic offensive language detection and hate speech detection on user-generated dialectal Arabic content.

Offensive (Abusive) Language

Providing a discrete definition of offensive (abusive) language is a very complex task. Culture and personal experience are crucial factors in describing what is considered offensive language (Vandersmissen, 2012). Wiedemann et al. (2018) describe offensive language as "threats and discrimination against people, swear words or blunt insults"

(p.1). Hate speech, aggressive content, cyberbullying, and toxic comments are different forms of offensive language (Schmidt & Wiegand, 2017). Some of these forms are discussed in more detail below. The use of offensive language can cause disturbance, affecting online harmony. Moreover, it can reduce user trust in the online platform (Fountaet al., 2018). Figure 1 shows an example of an offensive tweet in Arabic.

Figure 1 Example of offensive tweet

Hate Speech

Text that is targeted towards a group of people with the intent to cause harm, violence, or social chaos is known as hate speech (Sigurbergsson & Derczynski, 2019). Davidson et al. (2017) define hate speech as "a language that is used to expresses hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group" (p.1). Typical forms of hate speech usually contain racial and homophobic slurs (Davidson et al., 2017). Fortuna and Nunes (2018) define a list of rules to identify hate speech, which consists of: using stereotypes to refer to individuals based

on the groups to which they belong, using negative statements about minority groups, using racial and disparaging terms to cause harm, using racial and sexist slurs, using language that shows pride in a specific group, supporting entities that encourage hate speech, discriminating based on nationalities or religions, and using language that suggests the superiority of in-group individuals. Figure 2 shows an example of a hate speech tweet written in Arabic.

Figure 2 Example of hate speech tweet

Cyberbullying

Generally, cyberbullying is characterized by online harassment against an individual. Cyberbullying can have more severe effects than physical or verbal bullying, owing to the nature of online materials that can spread the harassment faster and make it viewable to a wider audience (Dadvar et al., 2013). Zampieri et al. (2019) define cyberbullying as targeted insults or threats toward an individual. Van Hee et al. (2015)

provide guidelines for analyzing cyberbullying; they mention three indicators of cyberbullying, including the intention to cause harm, repetitiveness, and an imbalance of power. Haidar, Chamoun, and Serhrouchni (2017) define eight categories for cyberbullying: flaming, masquerade, impersonation, harassment, outing, trickery, exclusion, and cyberstalking.

Adult Content

Adult content can take various formats in which the promotion of adult or sexual products, services, or content is expressed using images, videos, or text (Alshehri et al., 2018). Adult content also includes posts with vulgar content that contain explicit and rude sexual references and pornography (Mubarak et al., 2017).

Violence

Johnston and Weiss (2017) define violence in terms of terrorist groups (e.g., al-Qaeda and Islamic State of Iraq and Syria (ISIS)) and extremist groups (e.g., radical leftist (Antifa), White Supremacist). In (Abdelfatah et al., 2017), seven classes of social media violence are mentioned, including crime, violence, human rights abuse, political opinion, crisis, accidents, and conflict.

Motivation and Problem Definition

Studies have shown that cyberhate, online harassment, and other misuses of technology are on the rise. Particularly during the global Coronavirus pandemic in 2020, 35% of individuals who are part of a marginalized group reported online harassment related to their target's protected characteristics, which is a 3% increase over 2019.[1]

[1] https://www.adl.org/

Automatic offensive language detection has recently drawn much attention, as demonstrated by multiple recent competitions and shared tasks (Kumar et al., 2018; Bosco et al., 2018; Fersini et al., 2018; Wiegand et al., 2018; Basile et al., 2019; Zampieri et al., 2019; Mandl et al., 2019; Mubarak et al., 2020; Zampieri et al., 2020; Kumar et al., 2020; Fersini et al., 2020; Vu et al., 2020; Sanguinetti et al., 2020). However, automatic offensive language detection on social media is challenging because of limitations in understanding and interpreting the writer's intention and context (Nobata et al., 2016; Lees, Sorensen, & Kivlichan, 2020), as well as the informal, user-generated nature of online text(Pitsilis, Ramampiaro, & Langseth, 2018; Kapoor et al., 2019).

These challenges can be compounded by language specific ones. So far, research in automatic offensive language detection in Arabic has been limited. Arabic is a diverse language that has been polarized due to political and social conflicts, resulting in complex contextual information that can affect interpretation of offensive text. Additionally, Arabic is diverse in terms of its official forms; classical Arabic, modern standard Arabic, and dialectal Arabic (Habash, 2010). It also suffers from lack of widely available linguistic analysis tools (Haidar, Chamoun, & Serhrouchni, 2017; Mubarak, Darwish, & Magdy, 2017). Moreover, the ambiguity created by the variation among Arabic dialects directly effects identifying sub-culture specific offensive language.

Previous attempts at automatic Arabic offensive language detection follow the pipelines previously applied to English, failing to solve the problem inclusively and comprehensively. Most of these solutions also utilize a single dataset from one platform in training and evaluating their studies, potentially creating biased systems (Alakrot,

Murray, & Nikolov, 2018; Albadi, Kurdi, & Mishra, 2018, 2019a; Haidar, Chamoun, & Serhrouchni, 2017, 2018, 2019; Mohaouchane, Mourhir, & Nikolov, 2019; Mubarak, Darwish, & Magdy, 2017; Chowdhury et al., 2019; Aljarah et al., 2020; Djandji, Baly, & Hajj, 2020).

In contrast to the previous automatic Arabic offensive language detection studies, I propose methods that apply SOTA language models and consider multiple perspectives; text preprocessing, dialectal text, source platforms, and cultural aspects. To cover multiple Arabic dialects, I use nine Arabic offensive language datasets that cover diverse dialects, themes, and platforms.

Challenges

This dissertation concentrates on the following four challenges:

- Lack of structure in social media language:

The textual content of a social media post is not limited to words only; a collection of symbols, numbers, and representations of facial expressions and even objects, i.e., emoticons and emojis. For example, ":-D" emoticon refers to a happy face and "8-o" refers to a surprised face. 😂 emoji refers to a laughing face with tears and 🌹refers to a red rose. Emojis and emoticons are often used to convey feelings and attitudes, and can provide context for interpreting the tweets. Figure 3 shows an example tweet whose textual content, i.e., content consisting solely of Arabic words (e.g., alphabetical characters) is not offensive but whose interpretation in light of the emojis makes it offensive.

Figure 3 Example of a tweet that lack in structure

- Cultural, social, and political variations:

Arabic is the official language of most countries in the Middle East, which has been polarized during the last decades by several political and social conflicts. Detecting offensive language in this context requires knowledge about the cultural, social, and political situation of the post's source (e.g., author demographic), which may not be readily available for automatic methods (Çöltekin, 2020).

- Inconsistency among Arabic dialects:

The Arabic-speaking countries are widely spread in the Middle East region. Each country has its dialect and, in some countries, there are multiple sub-dialects. For instance, in the northern area of Saudi Arabia, the knife is usually called "Khousa/خوصه ", while in the middle area of Saudi Arabia, it is usually called "Sikkeen/ سكين". In general, all Arabic dialects could be grouped into seven main dialects: Egyptian, Levantine, Gulf, North African, Iraqi, Yemenite, and Maltese (Habash, 2010).

Social media posts are often written in dialectal Arabic and presents challenges for automatic offensive language detection across dialects. For example, the word cat in Egyptian is "Otta/أطة", in Levantine, it is "Bisse/بسة " in Gulf, it is "Qatwa/قطوة", in Moroccan, it is "Qetta/قط", in Yamani, it is "demah/دمة", and in Iraqi, it is "Bazzuna /بزونة". Moreover, it is very likely to observe contronyms in which the same word might have contradicting meanings. Such as the word "Afiah/عافية" which means "health" in Gulf, Egyptian, Iraqi, and Levantine and often be used in not offensive context, versus "fire" in Moroccan often be used in offensive context.

- Corpus dependency:

Automatic offensive detection can be corpus dependent and fail to generalize. For example, if the corpus used in developing the automatic method is biased towards specific Arabic dialects, then it may behave poorly when deployed with other Arabic dialects. Similarly, the corpus may be biased towards a specific social media platform. For example, some online platforms might have specific characteristics (e.g., users' demographic background, the country used) that affect the language used in its content, and might affect generalizability of the automatic detection system to other platforms.

Scientific Problem Description

Our goal is to automatically identify if a given short informal user-generated text contains offensive language or hate speech. Thus, we are given a set of user-generated short text posts, each with a class-label; offensive or not offensive, hate speech or not hate speech. From a training dataset of labeled samples, we set out to train a classifier that when receiving a new posting from a given user, can extract and combine the

information in the training dataset to classify the new posting into its appropriate class successfully. This type of problem is defined as a supervised text classification. Thus, given a set of text samples $S = \{ S_1, S_2, S_3 \dots S_n \}$, with their actual classes (labels) $C = \{ C_1, C_2, C_3 \dots C_n \}$, define a function F that takes a sample S_i and predicts its actual class C_j.

Research Questions and Goals

The scope of this dissertation focuses mainly on the following research question:

"How to effectively identify the offensive language of a new posting written in dialectal Arabic, given a set of known offensive and not offensive posts?"

To address the above research question, I:

1. Implement a set of **contextual-aware preprocessing** tasks to clean and normalize the text.
2. Create a **dialectal representation** that can accurately distinguish samples among different classes for Arabic text.
3. Develop a **classification model** that builds on the state of the art automatic offensive language detection in Arabic.

My hypotheses are as follows:

H1. Contextual preprocessing can improve automatic offensive language detection. Context from the perspective of this approach includes elements in the posts that are not part of any Arabic word and can enrich understanding of posts' content,

excluding pictures and videos. Thus, it includes emoji, emoticon, user mention, URL, newline, etc.

H2. Creating a dialectal representation can boost automatic offensive language detection.

H3. Developing a SOTA classifier based on a dialectal representation increases the accuracy of detecting offensive language in multi-dialectal and dialectal corpus.

Dissertation Contributions

Contributions of this dissertation can be summarized by the following:

- Advanced language models can capture the ambiguous, informal variations in the user generated text and accurately recognize offensive language.
- Different Arabic dialects can inform each other and significantly contribute to cross-dialect offensive language detection.
- Different user-generated content platforms do not add value to offensive language detection across platforms.
- We can customize existing SOTA language models to improve their coverage of dialectal Arabic.
- Dialectal Arabic language model outperforms the non-dialectal model on some offensive language detection datasets.

This dissertation has also contributed to multiple shared task submissions and academic research publications related to offensive language and hate speech detection. The work in this dissertation primarily relates to the following peer-reviewed articles (in order of publication):

1. **Husain, F.** (2020). OSACT4 Shared Task on Offensive Language Detection: Intensive Preprocessing-Based Approach. *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pages 53–60 with a Shared Task on Offensive Language Detection*. Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020.
2. **Husain, F.,** Lee, J., Henry, S., & Uzuner, O. (2020). SalamNET at SemEval-2020 Task 12: Deep Learning Approach for Arabic Offensive Language Detection. *Proceedings of the International Workshop on Semantic Evaluation 2020 (SemEval 2020).* The 28th International Conference on Computational Linguistic (COLING 2020), Barcelona, Spain, 12-13 December 2020.
3. **Husain, F.** & Uzuner, O. (2021). A Survey of Offensive Language Detection for the Arabic Language. The ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP).
4. **Husain, F.** & Uzuner, O. (2021). Fine-Tuning Approach for Arabic Offensive Language Detection System: BERT-Based Model. IEEE International Conference on Computer Applications & Information Security (ICCAIS'2021). March 18-20, Tunisia.

5. **Husain, F.** & Uzuner, O. (2021). SalamREPO: an Arabic Offensive Language Knowledge Repository. IEEE International Conference on Computer Applications & Information Security (ICCAIS'2021). March 18-20, Tunisia.
6. **Husain, F.** & Uzuner, O. (2021). Leveraging Offensive Language for Sarcasm and Sentiment Detection in Arabic. Proceedings of the Sixth Arabic Natural Language Processing Workshop. Association for Computational Linguistics. the 16th conference of the European Chapter of the Association for Computational Linguistics (EACL 2021).
7. **Husain, F.** & Uzuner, O. (2020). Investigating the Effects of Preprocessing Arabic Text for Offensive Language Detection and Hate Speech Detection Systems. The ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP).

The following preprints are also discussed:

1. **Husain, F. (2020).** Arabic Offensive Language Detection Using Machine Learning and Ensemble Machine Learning Approaches. arXiv: 2005.08946.
2. **Husain, F.** & Uzuner, O. (2021). Exploratory Arabic Offensive Language Dataset Analysis. arXiv:2101.11434

Finally, while not directly related, the following articles have also been completed throughout the Ph.D.:

1. **Husain, F.** (2020). Investigating Current State-of-The-Art Applications of Supportive Technologies for Individuals with ADHD. arXiv:2005.09993
2. **Husain, F.** & Motti, V. (2020). Social Media Usage in Kuwait: A Comparison of Perspectives Between Healthcare Practitioners and Patients. arXiv:2005.07898

Dissertation Structure

This dissertation is organized into 11 chapters. Chapter 2 explains the essential characteristics of the Arabic language required to understand the approach and methods used in the dissertation. Basic concepts are explained in Chapter 3. Chapter 4 presents a detailed survey of automatic Arabic offensive language detection methods and resources, highlighting the gaps, limitations, and findings from previous studies. Chapter 5 presents an investigation of the content and context of nine publicly available Arabic offensive language datasets used in conducting experiments in this dissertation.

The first attempt to develop an Arabic offensive language detection system using an intensive preprocessing-based approach is presented in Chapter 6. Chapter 7 focuses on transfer learning approaches and advanced language models for word representation with experimental studies to arrive to the best hyperparameters settings. The second attempt to develop an Arabic offensive language detection system is proposed in Chapter 8 based on transfer learning across several Arabic dialects. In chapter 9, the third attempt to construct an offensive language detection system is discussed, which applies transfer learning principles to several user-generated content platforms.

An approach to a customized SOTA contextual word representation model using cultural and dialectal dataset is explained in chapter 10. Chapter 10 also discusses the last attempt to construct a system for Arabic offensive language detection that utilizes the findings across all previous chapters and enriches the model with offensive dialectal and cultural knowledge. This chapter provides an evaluation of the proposed system using commonly applied statistical performance metrics. Chapter 11 provides conclusions and future research directions for the work of this dissertation.

THE ARABIC LANGUAGE

Amir al-Mu'minin Imām ‘Alī ibn Abī Tālib (peace and mercy be upon him) said:
" ما أفحَشَ كريمٌ قَطُّ."
“A person of dignity would never use obscene language”

- Ghurar al-Hikam, number 9478; Mizan ul Hikmah, page 646

The previous chapter introduces the scope of this dissertation and the problem of online offensive language detection for the Arabic language from a broad angle. It is very important to consider the characteristics and properties of the Arabic language when designing a solution for Arabic text as that can directly impact the taken procedures taken. The main objective of this chapter is to introduce the Arabic language. This chapter highlights some essential characteristics of Arabic, including dialectal Arabic. It includes examples from Arabic text to illustrate the challenges associated with the language. The next chapter shifts the topic toward text classification systems.

Arabic Language Background

Approximately 300 million people worldwide use Arabic as their primary language (Haidar et al., 2017). Arabic is the official language for the majority of countries in the Middle East region. It is also the original language of the Quran and the Hadith; thus, it is widely studied by Muslims around the world.

Arabic is part of the Semitic language family. Scripts in Arabic are written in the opposite direction to English scripts; thus, Arabic text is read and written from right to left. The Arabic alphabet consists of 28 letters. It has only two vowels, "أ/alif" and "ي/yaa". Arabic has two genders for nouns; feminine and masculine. The base form is the same as the masculine form. For example, the word "Qetta/قطة" refers to female cat and the word "Qett/قط" refers to male cat. In addition to the gender distinction in nouns, Arabic has singular, dual, and plural forms of nouns, verbs, pronouns, and adjectives. For instance, if we want to use the word cat to refer to two female cats in Arabic, the word "Qettatan/قطتان" is used, for two male cats, the word "Qettan/قطان" is used, and for plural cats, the word "Qettat/قطط" is used.

Arabic words are justified by using elongation or Tatweel, also called Kasheeda. Figure 4 shows an example of the use of elongation in Arabic words.

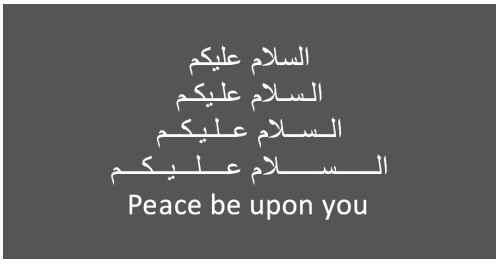


Figure 4 Examples of different sizes of elongation in Arabic words

There are different forms of Arabic: Classical Arabic Language (CAL) is the oldest and is often used in the Islamic manuscripts (e.g., the Quran); Modern Standard

Arabic (MSA) is the official language for Arabic countries and is in use in schools, media, books, etc.; and Arabic dialects are the native language form of daily communication. The Arabic dialects can differ in form, primarily based on geography and social classes (Habash, 2010). Moreover, Arabic dialects are often used in user-generated content such as Twitter, Facebook, and Instagram. Habash (2010) divides the Arabic dialects into the following seven categories:

1. Egyptian Arabic covers Egypt and Sudan.
2. Levantine Arabic covers Lebanon, Syria, Jordan, Palestine, and Israel.
3. Gulf Arabic covers Kuwait, United Arab Emirates, Bahrain, Qatar, Saudi Arabia (wide range of sub-dialects), and Oman (sometimes included).
4. North African Arabic covers Morocco, Algeria, Tunisia, Mauritania, and Libya (sometimes included).
5. Iraqi Arabic is a mixture of both Levantine and Gulf.
6. Yemenite Arabic is often considered its own dialect.
7. Maltese Arabic, which is not always considered one of the Arabic dialects.

Challenges of Processing the Arabic Language

Multiple Letter Shapes

The letters in Arabic can be written in multiple shapes depending on the letter's location within the word. For example, the letter "ف/faa'" can take the shapes "ـفـ/ فـ / ف" based on whether it is located at the beginning, in the middle, or at the end of the word (Gayar and Suen, 2018; Habash, 2010). Figure 5 shows an example.

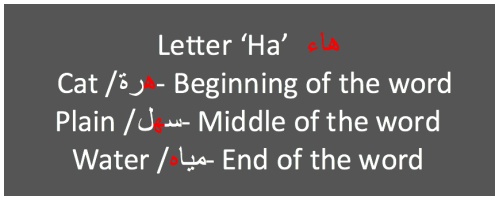


Figure 5 Variations in the shape of Arabic letters based on the location of the letter within the word

Ambiguity

Multiple words can have the same spelling but have different pronunciations and meanings depending on diacritical marks and punctuation. Diacritics are often called Tashkīl or Harakāt in Arabic, and they are used as a phonetic guide. Most available Arabic text is written without these marks, which creates ambiguity. For example, the Arabic word " شْعرٌ " ,"شَعْرٌ" means hair, " شَعَرَ " means he felt, and " شِعْرٌ" means poetry (Alhumoud et al., 2015; Boudad et al., 2018). Figure 6 shows an example.

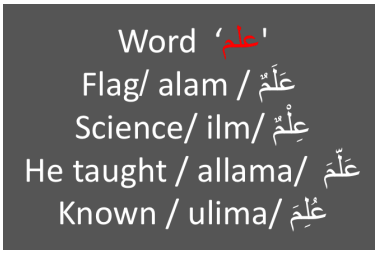


Figure 6 The effect of diacritics on the meaning of Arabic words

Ambiguity in Arabic can also be contextual. For example, the word " قتل/killing" in Arabic can be used in violent contexts as well as non-violent contexts, as it is in this

statement "تستطيع قتل الازهار ولكن لا تستطيع أن تمنع قدوم الربيع," which means "You may kill the flowers but cannot prevent the arrival of spring" (Abdelfatah et al., 2017).

Diversity of Arabic Dialects

The Arabic world's official language is the MSA, but the spoken Arabic is the dialectal Arabic (Habash, 2010). There are multiple Arabic dialects that vary among different regions and even within the same country (Alhumoud et al., 2015; Habash, 2010). Multiple dialects may share some words with the same spellings and pronunciations, while their meanings may differ. For instance, the word "ناصح/ Nasih" means "overweight" in Levantine, "smart" in Egyptian, and "advisor" in Gulf (Albadi et al., 2019b).

Diversity of Arabic User-Generated Content

Arabic text in user-generated content is more diverse than Arabic dialects. For example, Arabic text used in social media is usually the informal dialectal form of Arabic, including local words and slang words that differ among regions within Arabic-speaking countries (Abozinadah, 2017).

Posts in user-generated content are often written in Latin characters representing Arabic words in a transliterated form called Arabizi, Aralish, Arabish, or Franco-Arab (Darwish, 2014). Arabizi utilizes Latin letters, and in some cases, when the Arabic letter does not have some equivalent phonetic letters in English, it utilizes numeric and punctuation characters. For example, the number "2" represents the letter "أ" in Arabic (that sounds like "a" as in apple), and the number "3" represents the letter "ع" in Arabic (that is a guttural "aa'") (Darwish, 2014). Examples of words written in Arabizi are

“3a2albik/عالبك/on your heart” and “raw3a/روعة/wonderful”. Moreover, code-switching among the various forms of Arabic and sometimes English can be observed in the same post in user-generated content. Figure 7 demonstrates the diversity of user-generated content in the Arabic language.

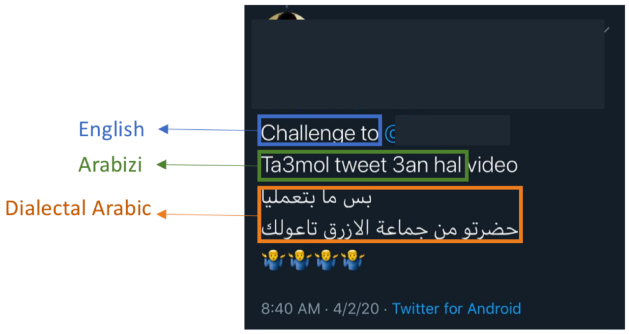


Figure 7 Example for a diverse user-generated post in Arabic

Some English terms are commonly used among Arabic speakers can be represented using Arabic letters, as shown in Figure 8.

Figure 8 Example English tweet written in Arabic alphabets

The Use of Urdu and Farsi Alphabets

In some Arabic dialects, the pronunciation of the Arabic alphabet differs. Thus, users use other non-Arabic languages to represent the dialectal sound of the letter to better describe the phonetic sound of the dialectal words. In this scenario, Urdu and Farsi alphabets are commonly used. Figure 9 shows an example of a tweet that is written in Iraqi Arabic, which replaces the letter "ك/kaa'" in Arabic with the letter "گ/Qa'" in Farsi and Urdu.

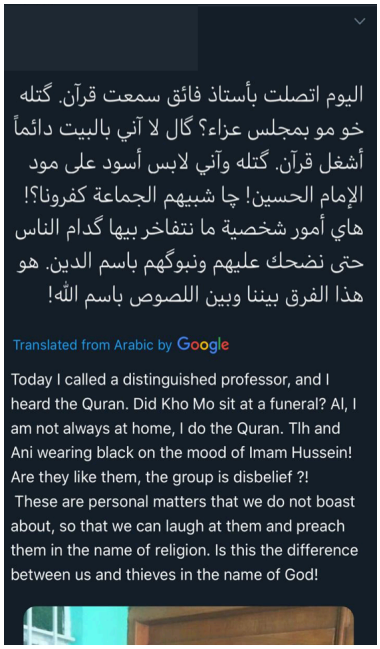


Figure 9 Example for the use of non-Arabic letters in dialectal Arabic

Chapter Summary

In this chapter, I shortly introduce the Arabic language to understand language-specific challenges that come in to play in developing effective Arabic offensive language detection methods. This introduction contains some general characteristics of Arabic and also discusses issues related to Arabic used in user-generated content. The next chapter broadly covers principal concepts related to offensive language detection.

BACKGROUND

Imām Jafar ibn Muhammad al-Sadiq (peace and mercy be upon them) said:
" معاشر الشيعة كونوا لنا زينا ولا تكونوا علينا شينا، قولوا للناس حسنا، واحفظوا ألسنتكم، وكفوها عن الفضول، وقبح القول."
"Be a source of honor and respect for us, not a means of disgust. Talk nicely with the people. Watch your tongue, and do not boast or swear."

- Bihar al-Anwar, volume 65, number 6, page 151; Mishkat ul-Anwar Fi Ghurar il-Akhbar, number 943, page 319

Before discussing the main content of my dissertation, I start with reviewing some background concepts to ensure a proper understanding of the next chapters. Thus, a person without a background in machine learning (ML) or deep learning can smoothly read all chapters. This chapter broadly explores the problem of offensive language detection from different perspectives, including those of a theoretical and practical nature. It covers the basic building blocks of building a system for automatic offensive language detection— such as challenges, methods, techniques, and algorithms— that are important to consider before developing a system in this domain. Additionally, this chapter includes a detailed preview for previous related research and studies that cover multiple types of offensive language from diverse languages, including English, German, Urdu, Turkish, Hindi, and Danish. A detailed survey of Arabic offensive language detection is provided in the next chapter.

Challenges

Previous research reports multiple challenges in working with user-generated online content for the purpose of analyzing offensive language. Besides the general difficulties of analyzing such user-generated text— such as informal language, misspellings, lack of structure, and slang— there are other challenges that are specific to offensive language detection systems. The following are some of those challenges:

Racial Bias

Kwok and Wang (2013) and Sap et al. (2019) examine the effect of social context (e.g., dialects, race) in detecting offensive language. They report racial bias during the process of textual annotation, in which a text that is labeled offensive by annotators is considered not offensive by people who are speaking the same dialect of the text (Sap et al., 2019). For example, the phrase "Wussup, n*gga!" is not considered offensive when used by the African American (AA) community, but is considered offensive if used by Americans of other ethnicities (Sap et al., 2019). Thus, it is critical to consider dialectal and racial differences in communication and language early in the study and account for them before beginning annotating procedures, in order to avoid unintended negative impacts (Sap et al., 2019).

Code-Switched Language Complexity

Code-switched language is common in multilingual societies, especially for social media content within offensive situations, which creates additional complexity during the process of offensive language detection. The spelling of code-switched words varies depending on the writer. Consequently, parsing this type of irregular text is challenging.

One study that investigates the use of Hindi and English, so-called 'Hinglish', on social media finds some difficulties in detecting offensive content. These difficulties are associated with the fact that Hinglish consists of concoctions of Hindi and English written in Roman scripts, which creates complex and unfixed spellings of text that make it hard for the algorithm to detect the meaning and semantics of multiple terms (Kapoor et al., 2019).

Dataset Limitations

Schmidt and Wiegand (2017) highlight some challenges of datasets used in offensive content analysis. These challenges are: a) variations in dataset sizes; b) uneven distribution of samples (most datasets tend to be highly imbalanced as far more not offensive samples exist than offensive examples); c) annotation of the datasets remains a very laborious task; d) platform specific characteristics. (There is a distinct difference in dialects between different platforms and demographics. A youth-focused social media platform will exhibit very different hate speech from a platform that has a more adult demographic.) The main lesson learned from previous studies regarding dataset consideration is the need to have a benchmark dataset for detecting offensive content, which helps in reducing bias among studies and creates more generalized models.

Obfuscation

It is a common practice for users to obfuscate offensive and abusive words in their posts, which makes it difficult for automatic systems to detect these words (Nobata et al., 2016). As a result, an online offensive language detection system based on an offensive word's list cannot work efficiently in identifying offensive content. For example, instead

of the word “nigger”, the words “ni99er” or “ni9 9er” are used. This problem of obfuscation creates challenges when features vectors are implemented, especially if pre-trained word embedding is used (e.g., GloVe, Word2Vec) that can reduce the performance of the classifier (Pitsilis, Ramampiaro, & Langseth, 2018).

Context Limitation

In some situations, it is important to look over the entire conversion, spanning multiple sentences, to accurately identify offensive content (Nobata et al., 2016). Not considering the context of the text can lead to false positive results. For example, a tweet that consists of “I am not sure if he is really on a low carb diet. Most hospitals do not carry low carb muffins LOL” can be misclassified as offensive while it should be not offensive. These issues come from the fact that there is no information present related to whether its sarcastic, whether the user is often offensive or sarcastic, or whether the culture views the tweet as such. Context-dependent text is difficult for automatic systems, or even humans, to classify correctly because it requires world knowledge related to the community of the platform and the users.

Datasets and Lexical Repositories

There are some public lexical resources that contain lists of offensive terms. These types of resources are often used to provide seed words for detecting offensive language. One of the largest hate speech resources is the Hatebase,[2] which contains over 3,632 terms covering 97 languages (Hatebase, n.d.). Despite having multilingual lists, culturally-specific terms are important to consider in detecting offensive language

[2] https://hatebase.org/

because what might be considered offensive in one culture might not be perceived as such by another. The PeaceTech Lab[3] lexicons avoid the problem of culturally-specific offensive terms by providing a series of hate speech lexicons based on countries undergoing political conflicts (PeaceTech, n.d.). In some studies, researchers use lists provided by lexical repositories as sources for the main keywords to extract from their dataset, which are then used to analyze offensive content from online platforms.

Davidson et al. (2017) uses the Twitter API to search for English Hatebase lexicons as the first step to extract tweets related to hate speech. From this initial dataset, they extract other tweets from the time-line of each user. Then, they randomly select 25,000 tweets that include words from the English Hatebase lexicons to be annotated by CrowdFlower (CF) workers. They use a labeling schema consisting of three classes to label each tweet: hate speech, offensive but not hate speech, or neither offensive nor hate speech. Most tweets were annotated by at least three coders, and an inter annotator agreement score was calculated to ensure the quality of the labeling. Many other researchers also use this dataset for offensive language research and report sufficient results (Kapor et al., 2018; Sap et al., 2019).

In addition to extracting datasets based on lexicon resources, there are multiple publicly available annotated datasets that may be used for the purpose of studying offensive language. This type of dataset allows researchers to bypass the effort of labeling the dataset. The Offensive Language Identification Datasets (OLID) are provided by OffensEval shared tasks, which are part of the SemEval shared tasks. The

[3] https://www.peacetechlab.org/

OLID English dataset from 2019 shared task 6 for offensive language detection contains a collection of 14,200 annotated English tweets organized via using a hierarchical annotation model. The top labeling schema consists of offensive or not offensive labels. Fine-grained labels for the offensive tweets contain sub-labels to categorize the type of offensives as targeted or untargeted. Then, more specific sub-labels are used to further categorize the targeted tweets into individual, group, or other. OffensEval 2020 expands the offensive language detection tasks by providing multiple offensive language detection datasets in several languages with similar labeling schema to the one used for the English dataset described earlier. Kaggle provides a toxic comments dataset, the "Toxic Comment Classification Challenge", which contains over 200,000 Wikipedia comments. Derczynski (2019) and Pelicon, Martinc, and Novac (2019) use the OLID 2019 in their studies to develop their offensive language detection classification model.

Bohra et al. (2018) construct the first hate speech online code-mixed dataset by using information from political events rather than lexicon resources or publicly available datasets. Their dataset consists of 4,575 tweets written in English and Hindi (Hinglish). The process starts with extracting tweets from the last five years using the advanced search properties of the Twitter Python API. Then, they use these tweets to identify specific hashtags and keywords for political events such as protests and riots. Tweets were filtered to remove all pure English language or pure Hindi language tweets. During the annotation process, two tasks were performed, the first task consisting of annotating each word to English, Hindi, or other (for symbols, emoticons, punctuation, named entities, acronyms, and URLs). The second task was to manually label each tweet as

either hate speech or normal speech. To ensure the quality of the dataset, Cohen's Kappa coefficient and inter annotator agreement were calculated. This dataset was used by other linguistic researchers in this domain as described on the next section (Kapoor et al., 2019).

Pre-processing

Cleaning and pre-processing text have direct effects on the accuracy score of the classification model. Different languages require different preprocessing approaches. In addition, text from social media datasets need to be processed using different methods distinct from other types of text. For an English social media dataset, text cleaning usually starts with removing HTML tags (Gitari et al., 2015). Word casing is usually standardized in some way. For example, some researchers convert all content to lowercase (Kwok & Wang, 2013; Davidson et al., 2017; Pelicon, Martinc & Novak, 2019), while others keep all uppercase words (e.g., USA) as they are but lowercase tokens beginning with an uppercase letter, followed by lowercase letters (e.g., Fairfax). Some researchers use word-level segmentation (Pitsilis, Ramampiaro, & Langseth, 2018; Pelicon, Martinc & Novak, 2019); however, that approach might have some limitations because the context of the text is not considered. Gitari et al. (2015) use sentence-level segmentation as the basic unit for analysis. Removing stop words and punctuation are typical strategies used in pre-processing text (Kwok & Wang, 2013; Pelicon, Martinc & Novak, 2019).

Stemming of a lexicon to its basic unit using Porter stemmer is used by some researchers to normalize the text by reducing the number of similar lexicons (Davidson et

al., 2017). Some researchers normalized the content to a specific sample length measured by number of words, especially for Twitter datasets. Founta et al. (2018) and Pitsilis, Ramampiaro, & Langseth (2018) limit the maximum size of each tweet to 30 words during model training, and padded tweets of shorter size with zeros. This preprocessing step is needed for some Artificial Neural Network (ANN) models, which is explained further in the following sections.

Furthermore, to ensure users' privacy, some researchers include in the preprocessing step the removal of personally identifying content. For example, username mentions are replaced by "@USER" and links to websites are replaced by "URL" (Wiedemann et al., 2018).

Feature Extraction

It is very difficult for a computer to understand text in its raw format. Text often comes in irregular structures, while an ML or ANN model needs to work with fixed formats of inputs and outputs. Therefore, text needs to be converted to numerical formats, specifically, to vectors of numbers. This process of converting words to numbers is called feature extraction. Previous researchers used several features in the pursuit of offensive language detection. In the following paragraphs, I discuss in detail some of the commonly used features employed in prior literature.

Bag-Of-Words

Multiple researchers use the Bag-of-Words (BOW) feature in their offensive classification model (Bohra et al., 2018; Derczynski, 2019). While the BOW feature is commonly used in computer vision because of its simplicity, in NLP applications it is

used when in-depth information about the words is not relevant to the task. The BOW model represents text as multisets, also called bags, without preserving the order of the words but keeping their frequencies. For example, if we have these two phrases: "I love this movie" and "I would recommend it", the BOW will indicate that "I" occurred two times in the phrases, and the words "love", "this", "would", "recommend", "it" and "movie" occurred once (Jurafsky & Martin, 2009).

While some researchers find positive performance by using a BOW approach for hate speech systems, BOW might be misleading. Kwok and Wang (2013) and Davidson et. al. (2017) report high recall score when implementing a BOW approach in their model, but a high rate of false positives was reported as well. Simply tracking the existence of offensive words without considering the context and culture of tweets results in a misclassification of hate speech (Kwok & Wang 2013; Davidson et al., 2017). Abozinadah and Jones (2017) and Abozinadah (2017) demonstrate how BOW does not work properly with short texts such as tweets in the Arabic language as well.

N-gram

An n-gram is a sequence of n words, thus a uni-gram is a sequence of one word, such as "please", "turn", "your", or "homework"; a 2-gram (or bigram) is a sequence of 2 words, such as "please turn", "turn your", or "your homework"; and a 3-gram (or trigram) is a sequence of 3 words, such as "please turn your", or "turn your homework" (Jurafsky & Martin, 2009). Davidson et al. (2017) use uni-gram, bi-gram, and tri-gram features to develop their offensive language classification model.

Another version of n-grams, character n-grams, consists of a sequence of n characters rather than n words. Character n-gram works better than word n-gram when repetition of letters within a word is commonly used like “thanks Joe!!! loooong time ago hahaha”. Bohra et al. (2018) use character n-grams where n varies from 1 to 3 characters in their classification model because their dataset consists of code-mixed language (English and Hindi). Character n-grams is language independent and can work with spelling errors, which is common in code-mixed texts. Nobata et al. (2016) also implement character n-gram with 3 to 5 characters to model the types of conscious or unconscious bastardizations of offensive terms.

Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (TF-IDF) is a form of sparse word embedding. It is used to compute the relative importance of a term in each tweet and the entire dataset by balancing among multiple words; (1) highly frequent, irrelevant words (e.g., the, a, that), (2) relevant words that frequently occurred nearby (e.g., sugar and apricot), and (3) irrelevant words that only occur once or twice (Jurafsky & Martin, 2019).

The first part of the TF-IDF score computes the normalized Term Frequency (TF) as the following:

Equation 1 Term Frequency

$$tf_{(term,tweet)} = \begin{cases} 1 + log_{10} count(term, tweet) & if count(term, tweet) > 0 \\ 0 & otherwise \end{cases}$$

The second part of the TF-IDF score computes the Inverse Document Frequency (IDF) using the logarithm as the following:

Equation 2 Inverse Document Frequency

$$idf_{(term)} = log_{10}\left(\frac{TotalNumberofTweets}{TotalNumberofTweets}\right)$$

The final weight is computed as the following for each word in each document:

Equation 3 Term Frequency-Inverse Document Frequency Weight

$$W_{(term,tweet)} = tf_{(term,tweet)} \times idf_{term}$$

The main advantage of TF-IDF is that it provides semantic relationship between words in the vocabulary, unlike BOW. On the other hand, TF-IDF provides sparse and large vectors for each word with lots of zeros. Some researchers use different levels of TF-IDF vectorization in their model, including words level, characters level, and n-grams level. Mustafa et al. (2017) and Davidson et al. (2017) use TF-IDF features in developing their offensive language classification model using Twitter datasets, while Wiedeman et al. (2018) report its unsuitability over Twitter datasets because tweets are very short, making it a suboptimal source for IDF estimations.

Sentiment Analysis

The process of understanding users' attitudes and emotions is called sentiment analysis. It has two main measurements. The first measurement is polarity, which gives a measurement of whether the statement is positive, negative, or neutral, and it has a floating number score within the range of -1.0 and 1.0. Positive statement has a value close to 1, while negative sentiment has a value close to -1. The second measurement is subjectivity, which gives a measurement of personal opinion content verses factual information, and it has a floating value within the range of 0.0 and 1.0. Statements closer

to 0.0 are more objective than statements with a value closer to 1.0. Derczynski (2019) uses the sentiment scores, calculated using the VADER[4] and AFINN[5] libraries, as a feature in his offensive language classification model. Gitari et al. (2015) also use a module for sentiment analysis, SentiWordNet,[6] and report some associations between offensive language content and the sentiment of the text.

Part-of-Speech Tagging

The syntactic structure of a sentence can also be used as a feature for detecting offensive language. Warner and Hirschberg (2012), Gitari et al. (2015) and Davidson et al. (2017) use the Part-of-Speech (POS) tagger and the standard Penn Treebank tag set to increase the accuracy score of textual analysis by considering the function of each token within the content. For example, frequent syntactic patterns that show noun and verb co-occurrence (e.g., kill and Muslim) with the POS tri-gram "DT Muslim NN" might be relevant for hate speech detecting systems.

Word2Vec

Word2Vec is similar to BOW in creating unique vectors for each word, and similar to TF-IDF in considering the semantic and context for each word. However, it creates dense vectors of short length and non-zero values, which generates more efficient features for NLP classifiers as it has less weight parameters to learn, fewer overfitting issues, and is able to capture synonymy (Jurafsky & Martin, 2019). For example, in a sparse vector embedding, the two synonyms "car" and "automobile" have two distinct

[4] https://pypi.org/project/vaderSentiment/
[5] https://github.com/fnielsen/afinn
[6] https://github.com/aesuli/SentiWordNet

dimensions and the similarity of the two terms as a neighbor is not captured, as it is the case in Word2Vec.

The software package of Word2Vec has two algorithms, skip-gram and Continuous Bag-Of-Words (CBOW), and is available with pre-trained embedding. The skip-gram works as follows (Jurafsky & Martin, 2019):

1. Use the target word and neighboring context words as the positive examples.
2. For the negative examples, randomly sample other words in the lexicon.
3. Train the classifier to differentiate between the positive and negative examples by using a Logistic Regression (LR).
4. Then, treat the regression weights as the embedding of words.

Therefore, the skip-gram model predicts the entire context from a target word. On the other hand, CBOW is very similar to skip-gram, but it predicts the target word given the context window words (Pennington, Socher & Manning, 2014; Dipanjan Sarkar, Apr 2018). Previous studies demonstrate that skip-gram gives better semantic accuracy than CBOW, and CBOW gives better syntactic accuracy (Mikolov et al., 2013). Figure 10 shows the architectures of both the skip-gram and CBOW models.

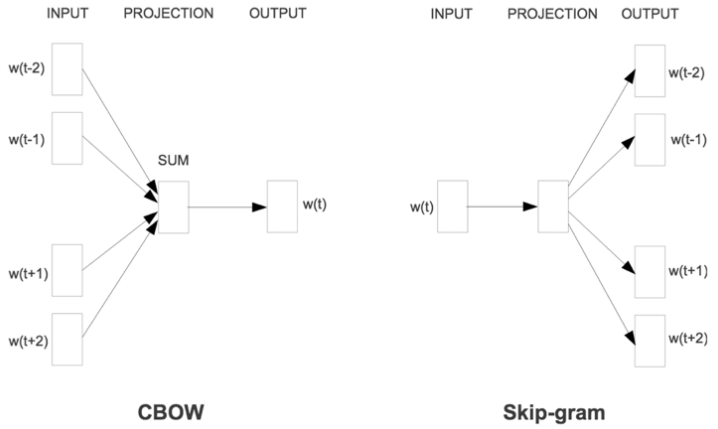


Figure 10 The difference between the CBOW model and the skip-gram model (Mikolov et al., 2013, p.5)

Paragraph2Vec, Comment2Vec, and Doc2Vec are extensions for Word2Vec. Derczynski (2019) and El-Zanaty, Aly, and Hossam (2019) use multiple word embeddings, including Word2Vec, in their offensive language classification model. Djuric et al. (2015) use the extended version of Paragraph2Vec, and Nobata et al. (2016) use Comment2Vec; both studies were for hate speech classification systems using a dataset consisting of Yahoo comments.

Global Vector

Global Vector (GloVe) is a word embedding method that combine the advantages of global matrix factorization and local context window methods. It uses the global word-word co-occurrence counts to train a weighted least squares regression model (Pennington, Socher & Manning, 2014). Similar to other unsupervised word representation methods, GloVe depends on the word occurrence statistics of a corpus. Pennington, Socher & Manning (2014) compare GloVe performance with other similar word embeddings, including Word2Vec with CBOW and Word2Vec with skip-gram;

they report a higher accuracy score when using GloVe than the others, and attributed that to the fact that GloVe depends on the entire corpus in identifying the context rather than a context window, as most of the other embeddings do. The findings of Pennington, Socher, and Manning (2014) findings are in line with those findings of Derczynski (2019) and El-Zanaty, Aly, and Hossam (2019).

FastText

FastText is an extension of Word2Vec that considers CBOW and skip-gram models in its word embedding. The modifications made were mainly in preprocessing strategies to support position dependency, phrase representations, and use of sub-word information (Mikolov et al., 2018). To support position dependency, a position-dependent weighting representation was implemented by learning position representations and applying them to re-weight the word vectors. In addition, instead of using uni-grams as in CBOW and skip-gram, FastText uses word n-grams to capture richer information. FastText uses the Word2Phrase tool from the Word2Vec software library to perform the task of phrase representation (Mikolov et al., 2018). Capturing sub-word information is especially critical for text with rare or misspelled words and for morphologically rich languages, which could be supported by decomposing each word into its character n-grams. This latter property of FastText makes it crucial for our particular task of classifying offensive language from social media sources because it provides semantic embedding for words that are not included in the training dataset during model training (Wiedemann et al., 2018). Mikolov et al. (2018) evaluate the performance of FastText on different kinds of corpora for text classification tasks and compare the results to GloVe.

Their results show slightly better accuracy with FastText, 82.7% for FastText and 82% for GloVe, which is consistent with findings from studies by El-Zanaty, Aly, and Hossam (2019) and Derczynski (2019).

User-Specific Features

Some researchers incorporate user-related features into their classification model. Dadvar et al. (2013) investigate the usage of user context to improve the detection of cyberbullying for a YouTube comments dataset. They define user context as a collection of features comprising: content-based features (e.g., number of profane words and number of first and second pronouns), cyberbullying features (e.g., the length of the comments and the number of cyberbullying words), and user-based features (e.g., history of user activity and age of user) (Dadvar et al., 2013). Their model reports a total lift of 9% when employing user-based context with a Support Vector Machine (SVM) based classifier (Dadvar et al., 2013). In another study, a classification model uses the users' tendency toward racism or sexism, measured by using their tweeting history, which shows an improvement in performance from an F1 score of 90.89% (textual content only) to an F1 score of 92.95% (with context) (Pitsilis, Ramampiaro, & Langseth, 2018).

Topic Modeling

Wiedemann et al. (2018) include in their feature extraction process an analysis for the usernames mentioned in tweets (e.g., @realDonaldTrump, @chrisbrown). They assume that offensive language is more likely to be used when the tweet is targeted to politicians or news agencies rather than to musicians or athletes. Their analysis includes tweets with at least two username mentions from their dataset, in which each username

mention occurred at least five times in the overall dataset. After that, they cluster user IDs for these username mentions using the Latent Dirichlet Allocation (LDA) algorithm for topic modeling, thus, creating thematic clusters that group similar accounts together (e.g., politicians, musicians, sport clubs). Each cluster has a distinct cluster ID that is later used as an input feature to the classification model. Wiedemann et al. (2018) demonstrate the positive effect of adding topic modeling features to the classification model on the performance score of their offensive language classification system, which uses a dataset of German tweets; the F1 score of 69.97% increases to an F1 score of 76.18% for the task of classifying tweets into offensive or not offensive classes.

Classification Models

Supervised ML is the predominant approach for classifying offensive language, with an SVM-based classifier being the most prevalent classifier and with a deep-learning-based classifier becoming prevalent in recent studies (Schmidt & Wiegand, 2017). In the following paragraphs, I review some information for ML and ANN models that were commonly used in previous studies for the task of offensive language detection:

Machine Learning Models

The main source of power for ML is encapsulated in its process of supporting the computer to be an active learner. Computers can teach themselves to use data and learn from their experiences to make more accurate decisions by using ML techniques. In ML, the data used in teaching the model is called training data, the data used for initial evaluation of the model's parameters is called the evaluation or development data, and the data used for testing the model is called testing data. The decision of an ML model is

either a classification decision or a prediction decision. There are three main learning algorithms used in ML models: (1) supervised learning in which the decision made by the model depends on using labeled data; (2) unsupervised learning in which the decision made by the model depends on analyzing similarities and differences between instances of data rather than using labels; and (3) semi-supervised learning in which both supervised learning and unsupervised learning are combined (Haidar, Chamoun, & Serhrouchni, 2017). In this research, I focus on supervised learning methods for offensive classification systems.

Jurafsky & Martin (2019) define four main components for ML classification systems: (1) a feature representation for each input instance that takes the shape of a vector of features $[x^1, x^2 \dots x^n]$; (2) a classification function to estimate the class for each instance ($\hat{y}$); (3) an objective function for learning through minimizing errors on the training dataset; and (4) an optimization algorithm for the objective function to generate the best output.

Ensemble ML methods are sometimes applied to enhance the results of ML classifiers by combining results (predictions or classifications) from a set of predictive models into one single prediction. There are five types of ensemble ML methods including boosting, voting, stacking, bagging (a.k.a. bootstrap aggregating), and random forest (Haidar, Chamoun, & Serhrouchni, 2017). The following paragraphs describe LR and SVM in detail, as they were commonly used in the previous offensive language classification studies, with SVM being the most dominant.

1. Logistic Regression (LR):

The LR is a discriminative classifier that can classify text into one of two classes (e.g., "Offensive", "Not Offensive"), or into multiple classes (e.g., "Offensive", "Hate Speech", "Neither"), in which case it is called a Multinomial LR. The LR is a supervised probabilistic classifier that requires a labeled training dataset to train the model. The goal of the LR model is to use the training dataset in training and developing a classifier, which can make a decision on the class of a new input instance. To achieve this goal, the LR learns from the training dataset a vector of weights (w) and a bias term (b), also called intercept. Both w and b are represented using real numbers associated with each one of the input features. The w is used as an indication of the importance of feature to the classification decision (Jurafsky & Martin, 2019). Learning procedures of the weights require a loss function (L) to identify how close the predicted classifier output is to the actual output for each instance. The cross-entropy loss function is commonly used to increase the actual class labels' probability for LR, which is called conditional maximum likelihood estimation. An optimization algorithm is applied to optimize weights given by the loss function. In LR, the gradient descent is often used as the convex optimization algorithm, which arrives at the minimum weights by identifying the direction that the function's slope is rising the most steeply, then the algorithm moves to the opposite direction (Jurafsky & Martin, 2019).

For language processing applications, the learning procedure of features can suffer from overfitting problems, requiring the addition of a regularization term (R(w)) to the objective function to make the model more generalizable. Two types of regularization

are commonly used in language processing including: (1) L1 regularization, also known as lasso regression, a linear function of the weight values that has a norm similar to Manhattan distance; and (2) L2 regularization, also known as ridge regression, a quadratic function of the weight values that has a norm similar to Euclidean distance (Jurafsky & Martin, 2019).

After training the model, the LR calculates the weighted sum of classes evidences for each instance in the testing dataset. The sum of weighted features can be represented using a dot product of two vectors. Prediction of the class depends on calculating the probability of events by using a logit function (Derczynski, 2019). This logit function is usually a Sigmoid function for a binary class classifier or a Softmax function for a multiple class classifier. The Sigmoid function has an S shape and takes a real value, which is mapped to the range of [0, 1]. It tends to squash outlier values to 0 or 1 because it is nearly linear around 0 and has a sharp slope toward the ends (Jurafsky & Martin, 2019). Figure 11 shows the Sigmoid function and shape.

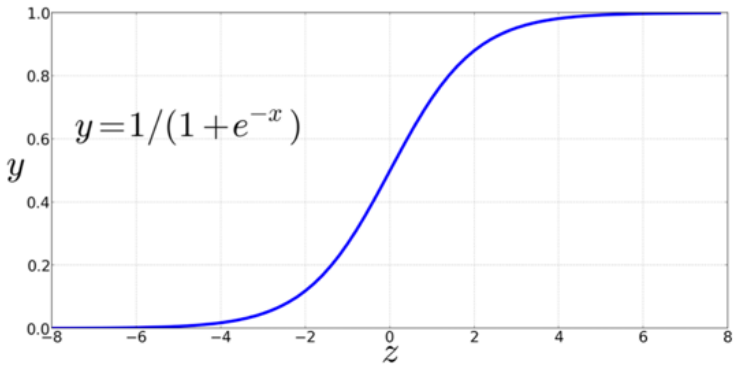


Figure 11 The Sigmoid function (Jurafsky & Martin, 2019, p.84)

The Softmax function is a generalization of the Sigmoid function. It takes the feature vectors and maps them to a probability distribution in which values are in the range of [0,1], and all add up to 1 (Jurafsky & Martin, 2019).

Previous studies deploy LR based classifiers for offensive language classification models, including Albadi, Kurdi, and Mishra (2018), Nobata et al. (2016), Davidson et al. (2017), Kent (2018), and Derczynski (2019).

2. Support Vector Machine:

Among the literature that I reviewed during this study, the SVM is the most frequently used classifier, specifically, within the Arabic language systems. Moreover, SVMs often report high accuracy scores among the literature. This finding is in line with that of the Schmidt and Wiegand (2017) study.

The SVM is a linear classifier that tries to find a separating hyperplane between instances of classes using a linear function. It has a constraint optimization problem; it looks for the hyperplane with the largest possible margin and does not provide probabilities. Thus, while the LR concentrates on maximizing data probability, the SVM focuses on maximizing the margin, the distance of the closest points to the separating hyperplane (Goodfellow, Bengio, & Courville, 2016).

In addition, the SVM can make effective use of large numbers of sparse features (Jurafsky & Martin, 2019). It can apply a kernel trick that makes computations of the final classification decision prediction very efficient (Goodfellow, Bengio, & Courville, 2016). The kernel transforms data into higher dimensional data, which supports more

accurate classification without the need to compute the exact transformation, by applying an inner product function to the data in the transformed higher dimensional space.

I discussed above some of the linear ML models that are commonly used in the domain of our study, detecting offensive language, based on previous literature. There are some other ML methods that are non-linear, such as the kernel method, decision tree, and boosting. The kernel method is a generalization method of the nearest neighbor algorithm, which depends on the label of the most similar instance from the training dataset to classify each new instance (Eisenstein, 2018). The decision tree method uses a set of conditions to classify a new instance (Eisenstein, 2018). The boosting method is similar to the ensemble method, which involves a combination of predictions from more than one "weak" classifier and using small subset of features (Eisenstein, 2018). Haidar, Chamoun, and Serhrouchni (2019) implement ensemble ML classifiers including stacking, bagging, and boosting methods in detecting cyberbullying content in Arabic tweets and report significant results with an F1 score of 92.6%. The next section discusses more complicated non-linear models than the ML non-linear models often used in deep learning.

Artificial Neural Network Models

As the name suggests, the ANN Model simulates the human brain to enable computers to make decisions in a human-like manner. The general architecture of the ANN Model consists of multiple layers of neurons, each having a computational unit with scalar inputs and outputs connected to each other to form a network (Goldberg, 2015). Inputs are associated with weights. The neuron sums the product of each input by

its weight. Then, it applies a non-linear function to the result and passes it to its output layer (Goldberg, 2015). The output of one neuron may feed into the inputs of one or more other neurons. The network can have multiple hidden layers between the input layer and the output layer, which are used by the classifier to enhance the process of learning and teaching the output layer the best function for the best output. Jurafsky and Martin (2019) describe the ANN model in very simple terms as a "series of logistic regression classifiers stacked on top of each other" (P.82). In other words, instead of teaching a computer to process and learn from the dataset as it is the case in ML models, in ANN models, the computer trains itself to process and learn from the dataset.

For some types of ANNs, the pre-processing of text needs to include the additional step of fixing the length of word sequence inputs. To mitigate the problems of large datasets, the ANN uses mini-batches in training the model in which every sample in a batch has the same sequence length, which can be set by the user. Another parameter for the ANN is the epoch, used to define the number of times the dataset gets cruised during model training. Multiple activation functions can be applied, including sigmoid, tanh, hard tanh, Rectified Linear Unit (ReLU), and its variants leaky ReLU and ELU. The same loss function of cross entropy used in LR is used for ANN as well, but computing the gradient of this loss function differs. The gradient is computed by an algorithm called Backwards differentiation, which depends on the chain rule in its computations (Jurafsky & Martin, 2019). In some cases, ANN makes use of a regularization factor during the training process to avoid overfitting, which is called the dropout rate. The dropout is used to randomly remove some units of the weights along with their connections from the

network. The value of this dropout factor is one of the network's hyperparameters that needs to be defined and tuned by the designer. Other hyperparameters may include: learning rate, mini-batch size, number of layers, number of hidden nodes per layer, and activation function (Jurafsky & Martin, 2019). The ANN has two parameters, the weights and the biases, which are learned by the gradient descent.

There are multiple models for ANN, and each works better for specific tasks. For example, the Recursive Neural Network has been proven to outperform other models in tasks with tree structures, like syntactic parsing (Socher et al., 2013). Meanwhile, the Convolutional Neural Network (CNN) and the Recurrent Neural Network (RNN) often work better in capturing the semantics of text and for text classification tasks (Lai et al., 2015; Wiedemann et al., 2018). I review some types of the ANN models which are commonly used for offensive language text classification tasks in the following paragraphs.

1. Convolutional Neural Network

The key advantage of using a CNN is its ability to extract meaningful salient features, such as tokens or sequences of tokens, in a way that is invariant to their position within the input sequences (Goldberg, 2015). It uses a max-pooling layer to determine discriminative phrases in a text (Lai et al., 2015). The pooling layer is an operation for down-sampling. Input to the CNN needs to be of equal length; zero-padding has usually been used to meet this requirement. Zero-padding is the process of adding zeroes to each side of the input's boundaries (Amidi, A. & Amidi, S., n.d.).

Figure 12 shows an example from Goldberg (2015) of a 1D convolution and pooling model for the sentence “the quick brown fox jumped over the lazy dog” using a window size of 3, in which each word is converted to a 2D representation vector. Then, those vectors are concatenated to result in 6D window representations. After that, a linear transformation filter of size 6 x 3 followed by element-wise tanh are used to convert each of the 6D seven windows to seven 3D filtered vectors. At the end, the max-pooling operation is performed to give the final 3D vector as shown in the figure below.

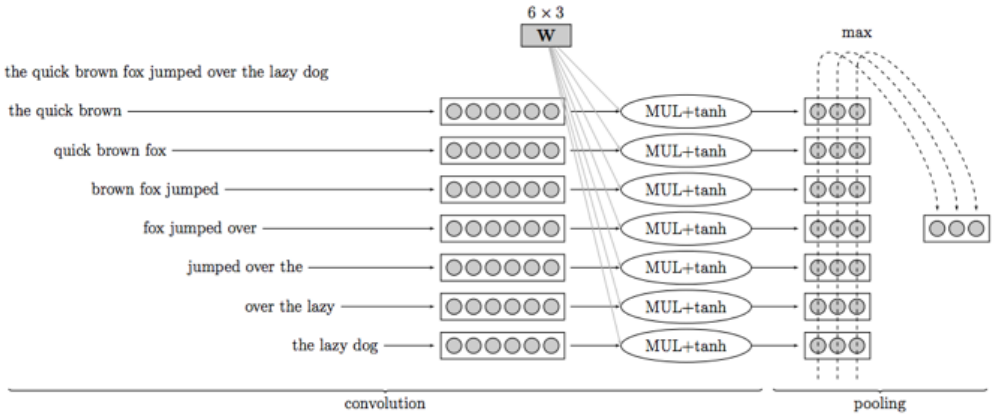


Figure 12 Example of a Convolutional Neural Network Model (Goldberg, 2015, p.44)

While the CNN works very well for data with grid-like typology (Goodfellow, Bengio, & Courville, 2016), the output representations from the CNN are very sensitive to the word order of input sequences. This order sensitivity is restricted to mostly local patterns, creating a bias model by disregarding the word order patterns that are far apart in the sequence (Goldberg, 2015).

Uglow, Zlocha, and Zmyslony (2019) develop a CNN model for an offensive language classification system using the OLID Twitter dataset. They report some difficulties in tuning the hyperparameters of the model until they were able to reach the best combination of values. For subtask a of classifying tweets to either offensive or not offensive, a F1 score of 0.735 was achieved, while for fine-grained classifications, task b and task c, lower F1 scores were reported (0.47 and 0.457 respectively) (Uglow, Zlocha, & Zmyslony, 2019).

2. Recurrent Neural Network

The RNN is distinguished by the addition of multiple loops to the ANN architecture. Those loops allow previous outputs to be reused as inputs while having a hidden layer (Amidi, A. & Amidi, S., n.d.). It works very well for data having sequential structure. The RNN can scale to very long input sequences of data with variable lengths, while the model size does not increase with the increase in input size (Goodfellow, Bengio, & Courville, 2016).

Founta et al. (2018) develop a unified architecture deep learning model based on RNN for their offensive language detection system trained on a Twitter dataset, and demonstrate its powerful generalizable performance by applying their model on another dataset from a different domain, toxic language in online video games, and report an accuracy of 0.93.

On the other hand, the computation of RNN is slow, and it suffers from difficulties with regard to accessing information from many steps back (Amidi, A. & Amidi, S., n.d.). In addition, the algorithm used during model training for the simple

RNN, the back propagation algorithm, which is used to compute the gradients and update the network weights in every layer, often faces a problem called the vanishing or exploding gradient phenomena. The vanishing gradient phenomena happens because, after a large number of back propagation steps, gradients tend to vanish. Consequently, omitting them is insignificant (Goldberg, 2015). To mitigate this problem of vanishing gradients, some variants of RNN use gates in their model architecture. The Gated Recurrent Unit (GRU) and the Long Short Term Memory Network (LSTM), which is a generalized form of GRU, are two variants of RNN. They are specifically designed to address the vanishing gradients issue by using different types of gates. The GRU uses an update gate and a relevance gate, while the LSTM adds to them two more, to include a forget gate and an output gate. Albadi, Kurdi, and Mishra (2018) develop a GRU model for investigating religious hate speech in Arabic tweets. They highlight its positive properties, including: GRU takes less training time, has superior performance on datasets with limited number of training instances, has better ability to generalize, and has less tendency to overfit with small datasets (Albadi, Kurdi, & Mishra, 2018). Derczynski (2019), Kapoor et al. (2018), and Pitsilis, Ramampiaro, and Langseth (2018) use the LSTM model for classifying different types of offensive language and report its usefulness for this domain of application.

In addition to the GRU and LSTM, the Bidirectional RNN (Bi-RNN) is another variant of RNN that is commonly used in NLP applications in which the whole input sequence is important. The Bi-RNN has two simple, separately trained RNN models, one model trained in the forward sequence direction, and the other trained in the backward

sequence direction, then, the outputs from both models are concatenated (Jurafsky & Martin, 2019). Derczynski (2019) uses the Bi-RNN model to classify Danish and English tweets in a multilingual offensive classification system.

Moreover, ANN can have augmentation with an attention mechanism. This attention mechanism has the ability to learn words relevant to the classification problem and assign them more weight. For example, in our task of offensive language classification, curse words will get higher weights than the other words (Mohaouchane, Mourhir, & Nikolov, 2019). Mohaouchane, Mourhir, and Nikolov (2019) use in their system a Bidirectional LSTM (Bi-LSTM) with attention mechanism, but it gives the least accuracy score among the other deep learning models they have created.

Goldberg (2015) points out the powerful outcome of combining multiple types of ANN in multiple layers within the classification system architecture to have better feature extraction and prediction at the same time. This combined approach was adopted by Lai et al. (2015) for general text classification using four multilingual corpora, and Wiedemann et al. (2018) for automatic offensive language text classification on a German Twitter dataset. Lai et al. (2015) create an RNN model that is able to capture contextual information by using the recurrent structure and, at the same time, can construct the text representation using the CNN. Wiedemann et al. (2018) develop a sequentially combined BiLSTM-CNN neural network, consisting of Bi-LSTM with 100 units followed by three parallel CNN layers using kernels of size 3, 4, and 5 with a 200 filter size. Results of their study report a higher accuracy percentage for coarse-grained offensive language detection, 77.5% compared to the 73.7% accuracy of the fine-grained

task of categorizing insults (Wiedemann et al., 2018). However, Albadi, Kurdi, and Mishra (2018) tried to develop an Arabic Tweets classifier for religious hate speech detection using CNN with GRU, and it did not offer better results than the GRU model alone. Mohaouchane, Mourhir, and Nikolov (2019) also investigate offensive Arabic language. They developed a combined model of CNN and LSTM, and it records the best results in terms of recall (83.46%) among the other deep learning models they explored during their study.

Performance Evaluation Measurements

There are multiple evaluation matrices and measurements for evaluating the performance of a classification model, each one focusing on specific attributes and having different weaknesses and biases. Thus, there is no "best" one because evaluating the model depends on its goal, task, and the domain of the application. We discuss some of the commonly used measurements for text classification in this section.

Confusion Matrix

The confusion matrix summarizes the performance of the classifier with respect to the testing dataset. Kai (2017) defines the confusion matrix as "a 2-dimensional matrix, indexed in one dimension by the true class of an object and in the other by the class that the classifier assigns". It can be used to evaluate binary classifiers or multiclass classifiers. Table 1 shows an example of a confusion matrix for evaluating a binary classifier with two classes: positive and negative.

Table 1 Binary classification matrix

Actual Class		Assigned Class	
		Negative	Positive
	Positives	False Negatives (FN)	True Positives (TP)
	Negatives	True Negatives (TN)	False Positives (FP)

As can be noticed from Table 1 above, the matrix includes four outcomes that can be defined as the following:

1. True Positives (TP) = the number of correct predictions that an instance is positive
2. False Positives (FP) = the number of incorrect predictions that an instance is positive
3. True Negatives (TN) = the number of correct predictions that an instance is negative
4. False Negatives (FN) = the number of incorrect predictions that an instance is negative

Moreover, multiple other evaluation measurements depend on these four outcomes, as we discuss in the following paragraphs. Botts (2019) highlights two points to consider when using the confusion matrix:

1. The full distribution of the data.
2. If the relevant class is very rare while the others are not, any significant amount of positive probability for it might be important to consider (e.g., spam detection, cyberbullying detection).

Abozinadah, Mbaziira, and Jones (2015); Haidar, Chamoun, and Serhrouchni (2017); Johnston and Weiss (2017); and Haidar, Chamoun, and Serhrouchni (2019)—along with multiple other researchers—use the confusion matrix in evaluating their offensive language classifier.

Contingency Table

Jurafsky and Martin (2019) define the contingency table as it is shown in Figure 13. It can be noticed that it is like the confusion matrix. Outcomes of the contingency table have the same definitions of those that the confusion matrix gives. The only differences are that the horizontal rows represent system predictions, while in the confusion matrix they are used for the actual label, and the columns are the actual labels in the contingency table while the columns are the system predictions in the confusion matrix.

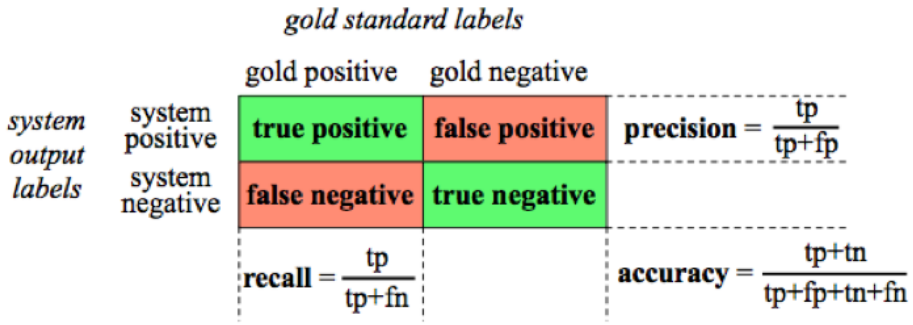


Figure 13 Contingency table (Jurafsky & Martin, 2019, p.74)

It should be mentioned that among all publications I reviewed for this study, we did not find any publication using the contingency table in their evaluation. However, we include it because Jurafsky and Martin (2019) in their book "Speech and Language Processing", report its importance as the starting step for NLP system evaluation.

Accuracy

Most of the literature uses an accuracy measurement to report the performance evaluation of their systems. Accuracy is a measurement of the proportion of the total number of correct predictions, in other words, the sum of the correct predictions divided by the sum of all predictions (Botts, 2019). The following function is used to compute accuracy based on previously defined confusion matrix outcomes:

Equation 4 Accuracy

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

Accuracy scores fall between 0 and 1, with 1 as the best accuracy. Botts (2019) defines two weakness for accuracy:

1. Accuracy gives one overall score, not per-class metrics.
2. Accuracy does not account for imbalanced datasets with unequal class distribution.

Thus, accuracy might not be a good choice of evaluation for our system if our dataset is unbalanced, as it is the case with most of the previous studies, since offensive content is usually less common than normal content in social media.

Recall

Recall is also called sensitivity or True Positive Rate. It measures the proportion of actual positives that are correctly classified by the system. Almost all previous works include recall in their evaluation metrics. The following function shows how to compute it:

Equation 5 Recall

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

Recall value's range is similar to accuracy. It falls between 0 and 1, with 1 as the best recall, however, it can be calculated per class. It might not be a valid choice for some systems if their main goal does not include capturing as many positives as possible. For example, recall is a valid measure for a cancer detection system since capturing the disease as early as possible is desired even if we are not very sure.

Precision

Precision also called Positive Predictive Value. It is the proportion of the predicted positive instances that were actually correct. The following function shows how to compute it:

Equation 6 Precision

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

Like recall, precision value ranges between 0 and 1, with 1 as the best precision. It can also be calculated per class. Precision is a good evaluation metric if the system needs to be very sure of its prediction, which is not the same case with regard to recall. There is a trade-off between precision and recall. One way to avoid this trade-off is by using the F1 measure as it is explained in the next paragraph.

F-Score or F1 Score

The F1 measure, also called F1 score or F-Score, is used to avoid the trade-off between recall and precision. It is the harmonic mean of both scores, and it incorporates the attributes of both. It has a similar range of values to them. It is the measurement that most previous work used in their classification system evaluation, since in most cases both precision and recall are equally important. The following function is used to calculate it:

Equation 7 F1

$$F1 = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

In addition to the above-mentioned evaluation measurements, we find that some studies apply different evaluation measurements to their model. Mustafa et al. (2017) develop multiple classifiers to detect controversial speech and patterns and use a paired t-test to evaluate the performance. Albadi, Kurdi, and Mishra (2018) use the Area Under Receiver Operating Characteristic (AUROC) curve to measure the performance of their Arabic religious hate speech classifier. Kent (2018) uses the Macro F1 measurement in

evaluating her German offensive language classifier. Alshehri et al. (2018) use the average F1 measurement to evaluate the performance of their Arabic adult content classifier.

Significant Test

The main goal of the statistical significant test is to evaluate if a model can be considered better than another one. It helps to quantify the probability that the difference between two models is not coincidental. Dror et al. (2018) provide a detailed guide to describe multiple significant tests and their usage in NLP. The appropriate significant test's selection depends on the distribution of the dataset; if the distribution is known, then parametric tests are recommended, while if the distribution is not known, then non-parametric tests should be used.

In NLP, the parametric significant tests include the Paired Student t-test, which assumes that both models' datasets come from normal distribution; thus, it examines if there is a difference between the population means of both models' measurements. On the other side, the non-parametric significant tests are applied when the test statistic distribution is unknown. Dror et al. (2018) classify the non-parametric significant tests in NLP into two main families: (1) sampling-free tests, which do not consider the evaluation measures' actual values; and (2) sampling-based tests, which consider the actual values of the measures by using the test data to evaluate repeatedly sample, and predict the significance's probability based on the test statistic values in the samples. In this dissertation, I apply the Paired bootstrap test, which is a sampling-based test that fits small data size and in which the sampling from test data is done with replacements. The

figure below shows the decision tree for statistical significance test selection according to Dror et al. (2018).

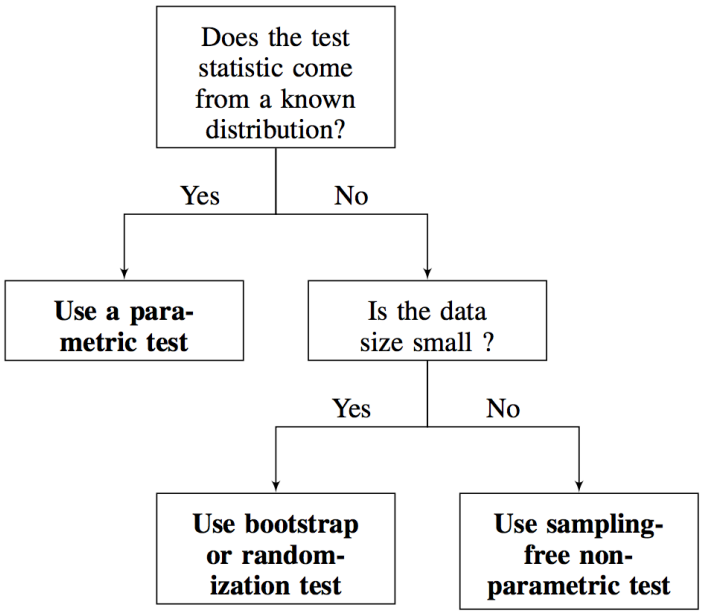


Figure 14 Decision tree for statistical significance test selection (Dror et al., 2018, P. 1388)

Adopted Approaches for Offensive Language Detection

Approaches used in offensive studies vary depending on multiple factors, including the languages of texts used in the datasets. In this section, I review some research approaches from previous studies based on the languages they used to develop their classification systems.

English Language Approaches

There is plenty of literature covering offensive language detection for online English language content. We select just few of them to discuss in this section. According

to the SemEval 2019 report for the offensive language detection task, it is apparent that deep learning based models are increasingly being used to help classify hate speech, with a Bidirectional Encoder Representations from Transformers (BERT) language model based approach performing the best (Zampieri et al., 2019).

Kwok and Wang (2013) focus on racial tweets against African Americans. As an initial step of the study, they did a survey using a hundred tweets that include keywords or sentiments of hate speech, then had three students of similar age and the same gender from different races label these tweets. The labeling schema has two classes, offensive or not offensive, and if a tweet was found to be offensive, they were rated on a scale from 1 to 5 (5 being the most offensive). Then, Fleiss' Kappa measurement was used to assess the reliability of agreements among coders. Results from this initial survey created labels explaining why each tweet was found racist. Then, a dataset of 24,582 tweets was collected using known racist Twitter accounts through reputable news sources. Preprocessing of tweets included removing URLs, mentions, stopwords, and punctuation, and lowercasing all words. In addition, converting alternative spellings of slurs to its properly spelled equivalent was done as part of the preprocessing step. The BOW was used to create the feature vectors. Kwok and Wang (2013) develop a Naive Bayes classifier and evaluate its performance using a 10-fold cross-validation method. The model shows an average accuracy of 76% and an average error rate of 24% (Kwok and Wang, 2013).

Davidson et al. (2017) develop a system that can distinguish between hate speech and other offensive language by using various ML classifiers. To pursue their goal, a

dataset was collected from Twitter using a list of hate words provided by HateBase, as discussed in detail earlier in the previous section. Tweets were labeled using three classes: hate speech, offensive language, or neither, and pre-processed by lowercasing and stemming. Multiple features were explored including: TF-IDF for bi-grams, uni-grams, and tri-grams; POS tag for uni-grams, bi-grams, and tri-grams; tweet quality by modified Flesch-Kincaid Grade Level and Flesch Reading Ease scores; sentiment scores; binary and count frequencies for hashtags, mentions, retweets, URLs, characters, words, and syllables in each tweet. The system has multiple classifiers such as LR, Naive Bayes, Decision Trees, Random Forests, and linear SVM. Each model was tested using 5-fold cross validation, holding out 10% of the sample for evaluation (Davidson et al., 2017). The results show an overall F1 score of 90% for the best classifier (Davidson et al., 2017).

Another study uses several classifiers based on the LSTM model to analyze hate speech, including racist or sexist content (Pitsilis, Ramampiaro, & Langseth, 2018). They used a publicly available dataset of 16,000 tweets with labeling classes: racism, sexism, or neutral. Their approach in tackling this task is one of the pioneering works with regard to the consideration of additional features related to users' tendency towards hateful behavior, captured from users' history and incorporated into their model (Pitsilis, Ramampiaro, & Langseth, 2018). For the textual features, they created a language independent feature to represent word vector by using word-based frequency vectorization, in which the words in the corpus are indexed based on the number of occurrences within the corpus. The index value for each word in a tweet is used as an

element to represent that tweet. Before extracting features, tweets were preprocessed to a fixed length of 30 words per tweet and tokenized, but punctuation was kept (Pitsilis, Ramampiaro, & Langseth, 2018). They used an ensemble of RNN classifiers and LSTM based classifiers, which reported a very high score in performance of F1; 93.20% (Pitsilis, Ramampiaro, & Langseth, 2018).

Other Monolingual Approaches

In a study for Urdu language tweets, a mixture of both ML and data mining techniques were used to detect controversial speech and patterns (Mustafa et al., 2017). Their definition of controversial speech is inclusive to both hate speech and cyberbullying, as they define controversial tweets as "those that contain abusive content that target individuals (e.g., a politician, a celebrity, or a product) or some groups (e.g., a country, a religion, a gender, or an organization)" (Mustafa et al., 2017, p.27). They used the Twitter search and stream API to collect tweets from official popular news pages and political pages during a period of six months, from December 2016 to June 2017. The size of the dataset, after filtering out all non-Urdu tweets, was 8,000 tweets. After extracting the dataset, they preprocessed tweets by removing: stopwords, URLs, replies, image-only tweets, and link-only tweets. Then, tweets were tokenized and TF-IDF for feature weighting was implemented. They used a manual annotation schema with two labels, true for controversial tweets or false for non-controversial tweets. Three ML classifiers were trained using the WEKA software program, including SVM, Naive Bayes, and LR, using 10-fold cross validation. The Naive Bayes classifier gave the best results with an F-measure of 0.89. Mustafa et al. (2017) also use some data mining

techniques for the instance with true labels (controversial) from the SPMF data mining library to find hidden patterns and relationships, including Top-K Sequential pattern mining (TKS), Co-occurrence MAP (CM-SPAM), and Equivalence class based sequential Rule Miner (ERMiner). Results for TKS with k= 1 and 2 words did not provide good indications of controversial content, while results for 3, 4, and 5 words TKS show some meaningful patterns, such as "نواب براہمدغ بگٹی" (Nawab Brahamdagh Bugti), and report some relationships with the Pakistan Army, Baluchistan, Terrorism, Shooting and Killing. Results obtained from applying CM-SPAM were very similar to those obtained from TKS. Last but not least, results obtained from ERMiner using a minimum frequency threshold of 1% and sequential rules confidence of at least 70% reveal relevant patterns such as "گالیاں <== افغان , مہاجرین ", which shows hate speech toward Afghan refugees living in Pakistan (Mustafa et al., 2017).

Özel et al. (2017) studies Turkish texts to build a system for the automatic detection of cyberbullying content in Instagram and Twitter posts. They manually extract and label a 900-message Turkish dataset from Instagram and Twitter, which consists of equal instances of both classes, "bullying" and "not bullying", each class having 450 instances. In addition, they consider the gender of the messages' author and ensure that within each class, they have an equal distribution for authors of both genders, for example, 225 male cyberbullying and 225 female cyberbullying. During dataset preprocessing, they created two versions of dataset. One version of the dataset was cleaned by removing all numeric characters and all punctuation marks, while the second version was cleaned by removing all numeric characters and all punctuation marks except

those from a specific, manually-defined emoticons list. Both datasets were converted to lowercase and split based on white space characters. Then, each message was converted to the BOW vector and the TF-IDF weighting vector. After that, they used WEKA to create several classifiers, including SVM, decision tree, Naïve Bayes Multinomial, and k-Nearest Neighbors (kNN). They also applied some feature selection methods such as Chi-Square (CHI2) and Information Gain (IG). Results reported a highest accuracy of 84% when using the kNN on the second version of the dataset that includes emoticons, and when applying IG feature selection. In general, applying feature selection improves the performance for all classifiers except decision tree, and IG shows improvement in results more than CHI2.

Kent (2018) builds a model for classifying a German Twitter dataset using a hierarchal labeling schema consisting of two levels. The first level labels tweets as offensive or other, and then on the second level, the offensive sample is further classified into three subtypes: insult, profanity, or abuse. The dataset contains 8,541 tweets adopted from the GermEval 2018 shared task. During the pre-processing step, all usernames were removed to anonymize tweets, then all punctuation was removed and Spacy lemmatizer was used to lemmatize texts. Several features were explored, including number of swear words, TF-IDF, n-grams, number of non-alphanumeric characters, and sentiment analysis (for text and emoji). Kent (2018) uses an LR based classification model with 10-fold cross validation, and the best performance achieved with character n-grams features had an F1 score of 76.72% for the first labeling level of tweets to offensive and others, and 47.17% for the further labeling of the offensive tweets to the three sub-types. The lower

F1 score for the second labeling level was a result of the problematic task of defining hate speech, which was hard for German annotators to perform as hate speech can be very subjective (Kent, 2018).

In another study for the German language, the researchers studied the effects of transfer learning on the performance of an offensive language detection system. Wiedemann et al. (2018) developed a sequentially combined BiLSTM-CNN neural network, consisting of Bidirectional LSTM with 100 units followed by three parallel CNN layers using kernels of size 3, 4, and 5 with a 200 filter size. A dataset of German Tweets was used to train and test the model. FastText provided sub-word embedding, which can better fit a Twitter dataset, as tweets are often written in informal language. To investigate the effect of transfer learning, they pre-trained the model using: (a) supervised class transfer— labeled user's comments with near-offensive language classes (inappropriate, discriminatory, or none); (b) weakly class transfer— labeled tweets by the emoji they contain; and (c) unsupervised class transfer: labeled tweets by the LDA for topic modeling (Wiedemann et al., 2018). In addition, they deploy three strategies to mitigate catastrophic forgetting during training on actual datasets, including gradual unfreezing, single bottom-up unfreezing, single top-down unfreezing, and no freezing of weights as their baseline strategy. Results report a best accuracy percentage for coarse-grained offensive language detection of 0.828 when pre-training the model with unsupervised class transfer for topic modeling using single bottom-up unfreezing strategy. For the fine-grained task of further categorizing offensive language, the highest accuracy percentage is 76.4%, which is identical for both models— pre-trained on

unsupervised class transfer for topic modeling and weakly class transfer of labeled tweets by emoji— when using single bottom-up unfreezing strategy (Wiedemann et al., 2018).

Multilingual Approaches

Kapoor et al. (2018) develop an LSTM based model to detect hate speech using two Twitter datasets, an English Twitter dataset (provided by Davidson et al. (2017)) and another one that contains Hinglish text (code-switched language of Hindi and English provided by Bohra et al. (2018). Both datasets have three labels: offensive, abusive, and none (or benign). After the basic preprocessing of text (e.g., removing stopwords, removing punctuation) and before developing the model, they used several methods to ensure the proper conversion of all texts to English: the Xlit-Crowd Hindi-English Transliteration Corpus, Hindi to English dictionary, a manually-created 7,200 word pairs dictionary of popular Hinglish words, and a manually-created profanity dictionary of 209 profane words (Kapoor et al., 2019). An example of a hate tweet before translation to English is “Bik gya Porkistan”, and after translating is “Porkistan (Derogatory term for Pakistan) has been sold” (Kapoor et al., 2019). They used word embedding trained with GloVe and Word2Vec on both datasets and used fixed embedding size of 100 (Kapoor et al., 2019). The LSTM based model has one LSTM layer with a 0.2 dropout value, and 3 dense layers. The last layer uses a categorical cross-entropy loss function, and Adam optimizer with L2 regularization was used to prevent overfitting. Grid search was used to select hyperparameters of the model. Results showed 87% accuracy for the model with GloVe and 82% accuracy for the model with Word2Vec (Kapoor et al., 2019).

In another multilingual study by Derczynski (2019), an offensive classification system for both the Danish and English languages was developed using ML and ANN classifiers. The OLID dataset described in the previous section was used for developing the English language system, and the same extraction procedures and annotation schema of the OLID dataset was used to construct an equivalent Danish dataset. However, Facebook and Reddit were used as sources as well, because Twitter has limited usage in Denmark. To ensure online users' privacy and to comply with the General Data Protection Regulations in Europe (GDPR), the following measures were taken: personally identifying content was replaced with @USER (e.g., usernames, without including celebrity names), and other sensitive information (including racial origin, political opinions, religious beliefs, philosophical beliefs, trade union membership, genetic data, and bio-metric data) was removed (Derczynski, 2019). Annotator agreement was measured using Jaccard index. Then, a set of features was implemented, including n-grams, word-to-index, sentiment score, reading ease (Flesch-Kincaid Grade Level and Flesch Reading Ease scores), POS tags, TF-IDF, FastText, Word2Vec, GloVe, and other linguistic features such as syllable count, character count, URL counts, and number of unique words. Derczynski (2019) developed four classification models for his system using LR, Learned-BiLSTM (the activation function for LSTM layers was tanh, ReLU for hidden layer, Sigmoid and Softmax for output layer), Fast-BILSTM (embedding layer initialized with FastText embedding), and AUX-Fast-BiLSTM to add more features to Fast-BILSTM. Grid Search Cross Validation was used to select the optimal parameters for the models, which end up with batch size of 128, optimization function of Adam with

learning rate of 0.001, and a dropout rate of 0.2 among all layers. Results for subtask A in which text was classified as offensive or not offensive reported best performance for the English dataset with Fast-BiLSTM having an F1 score of 73.5%, and for the Danish dataset reported best performance with LR having an F1 score of 69.9%. Results for subtask B in which offensive text was further classified to targeted insult or untargeted insult showed the English dataset's best performance when using Learned-BiLSTM, achieving an F1 score of 61.9%, and the Danish dataset's when using AUX-Fast-BiLSTM, achieving an F1 score of 72.9%. The last results for task C that includes further classification of the targeted insult sample to individual, group, or other, show best performance in both datasets using Learned-BiLSTM with an F1 score of 55.7% for English and 62.9% for Danish (Derczynski, 2019).

Chapter Summary

This chapter provides a preview of core concepts within the domain of text classification and offensive language detection systems. It starts with discussing the challenges of online offensive language detection systems, which include racial bias, code-switched language complexity, dataset limitations, obfuscations, and context limitations. Then, the chapter briefly reviews offensive datasets, lexical repositories, and preprocessing procedures. Multiple features that are commonly used for offensive language detection systems are explained, such as BOW, n-gram, TF-IDF, sentiment analysis, POS tagging, Word2Vec, GloVe, FastText, user-specific features, and topic modeling. The BERT model is discussed in detail in chapter 6. This chapter also discusses different types of classification models; LR and SVM for ML models and CNN

and RNN for ANN models. Most widely-applied performance evaluation measurements were reviewed, such as confusion matrix, contingency table, accuracy, recall, precision, and F1 score. It ends with examples of offensive language application approaches for multiple languages, including English, German, Urdu, Turkish, Hindi, and Danish. The next chapter provides an in-depth discussion for Arabic offensive language literature.

A SURVEY OF OFFENSIVE LANGUAGE DETECTION FOR THE ARABIC LANGUAGE

The Holy Prophet Mohammad (peace be upon him and his holy progeny) said:
" ما كانَ الفُحشُ في شيءٍ قَطُّ إلّا شانَهُ، ولا كانَ الحَياءُ في شيءٍ قَطُّ إلّا زانَهُ".
"No sooner does obscene language accompany something than it disgraces it, and no sooner does modesty accompany something than it adorns it."

- Bihar al-Anwar, volume 79, page 111, number 6; Mizan ul Hikmah, page 645

This chapter investigates in-depth automatic offensive language detection systems for Arabic from NLP perspective. It reviews previous studies on offensive language detection in Arabic, and presents a detailed investigation of the SOTA techniques covering this topic, pointing out open problems and limitations and showing gaps in previous research. It covers the following research questions:

- What are the available linguistic resources for Arabic offensive language? What functions do they serve, and what is their scope?
- What are the trends in Arabic offensive language detection literature?
- What types of offensive language are most commonly targeted?
- Which platforms are used as the source of data? What are the characteristics of the datasets?

- What methods and techniques are typically used for studying Arabic offensive language in user-generated content?
- What system performance and accuracy have been achieved in classifying Arabic offensive content?

I attempt to answer the above research questions by qualitatively and quantitatively analyzing 35 selected studies. Among the 35 studies, eight cover more than two categories of offensive content (e.g., hate speech and abusive language in the same study).

Research Methods

The scientific study of automatic offensive language detection in Arabic from NLP perspective is recent, and the number of studies in the field is low. In this survey, each study is considered as a unit of our analysis. I found only 35 studies during the process of the survey, which involved six main steps. I provide a precise description of each of these steps to support study replicability. The six steps start with keywords selection, followed by a search for studies, recursive search, filtering the studies, qualitative and quantitative analysis, and synthesizing data for identifying gaps in the literature. The steps are illustrated in Figure 15 and explained further in the following paragraphs.

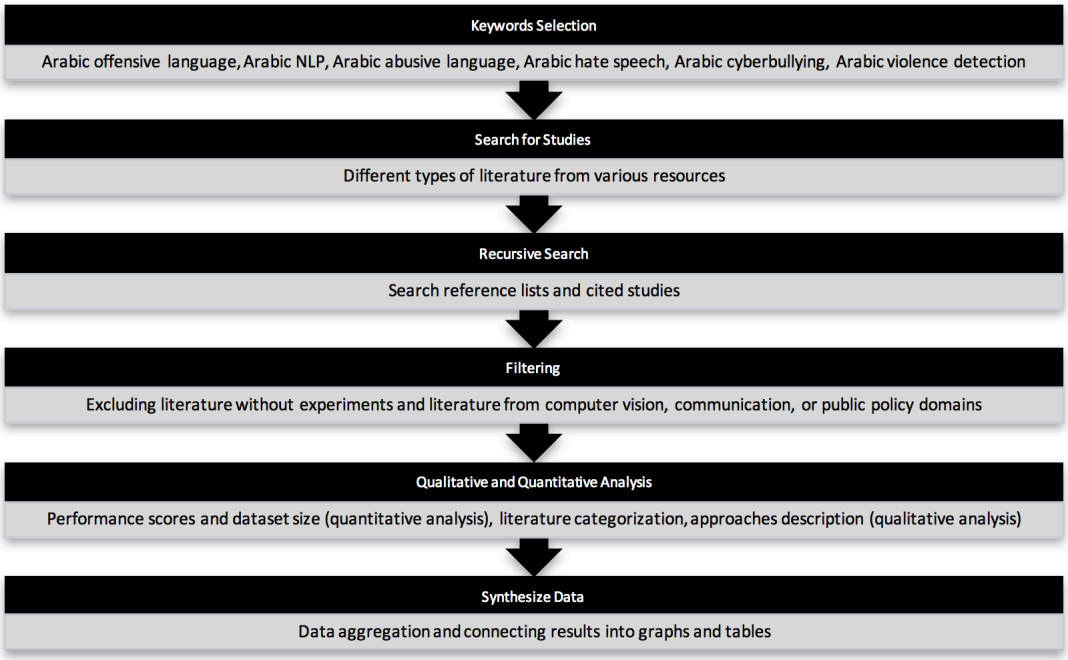


Figure 15 Steps of the Survey

Keywords Selection

Some keywords were selected during the first step of the survey. The keywords list includes "offensive language" and "Arabic NLP", which must be present anywhere in the text. Besides, I consider terms referring to particular categories of offensive language, "Arabic abusive language", "Arabic hate speech", "Arabic violence detection", and "Arabic cyberbullying", at least one of which must be present anywhere in the text.

Search for Studies

I aimed to collect the largest possible number of studies from the domain. Thus, I did not restrict the search to specific databases; rather, I started the search from the commonly used databases, such as the Association for Computing Machinery Digital

Library (ACM DL); the Institute of Electrical and Electronics Engineers (IEEE) Xplore; and the Association for Computational Linguistics (ACL). I included conference papers, workshop papers, conference posters, journal publications, dissertations, theses, and books.

Recursive Search

I used Research Gate and Google Scholar to retrieve the references listed in the studies found in the previous search and to retrieve other studies that cited the original work. Then, I recursively repeated the search one more time with the new studies found. The search stops after I am done with all literature published during the survey's time boundaries.

Filtering

To ensure the quality of the data collected, studies were filtered using some inclusion and exclusion criteria. The inclusion criteria cover a variety of publication types (e.g., conference publications, workshop papers, journal publications), collected from various resources (e.g., ACM DL, IEEE Xplore, ACL), which study several categories of offensive language (e.g., hate speech, cyberbullying, violence, abusive language), and utilize any Arabic language datasets.

The exclusion criteria limit the scope to the NLP domain (e.g., classification systems, linguistic resources, deep learning, machine learning, statistical learning). Moreover, studies that did not include any experiments were excluded. For example, the study "Detection of Hate Speech in Social Networks: a Survey on Multilingual Corpus" done by Al-Hassan and Al-Dossari (2019) was excluded from the survey because it does

not have an experiment or an implementation of a system. All studies published before June 2020 were evaluated.

Qualitative and Quantitative Analysis

Qualitative analysis examines the tools and resources used, the general approach, and the offensive language task tackled. Quantitative analysis also focuses on analyzing studies, for example in terms of the accuracy of the results obtained and the dataset size.

Identify Gaps

After analyzing literature quantitatively and qualitatively, I identify the limitations of previous research and highlight the future direction of research in Arabic offensive language detection.

Results

Qualitative Analysis of Studies

I analyze each study's content qualitatively to extract information related to Arabic linguistic resources, tools, and approaches followed by the studies.

1. Linguistic Arabic Language Resources and Tools

There are very few linguistic resources for the Arabic language (Haidar, Chamoun, and Serhrouchni, 2017). Arabic corpora, lexicons, and tools are very scarce, which limits research in this area. In addition, the available Arabic language analysis tools have some limitations in dealing with the complexity of the language (Mubarak et al., 2017). In this section, I discuss publicly available online resources and discuss in more detail both private and public resources in discussion section.

1.2 Datasets and Lexicons

In (Mubarak et al., 2017), Mubarak, Darwish, and Magdy create a MSA dataset to detect personal attack, racist, sexist, offensive, inciting violence, non-relevant, or advertising comments extracted from users' comments of an Arabic news outlet called Aljazeera.net. The initial dataset consists of 400,000 comments from a total of 10,000 articles that cover multiple themes: politics, economics, society, and science. After filtering the comments to include comments of three to 200 characters' lengths only, the final dataset shrinks in size to 32,000 comments. Then, three annotators from CrowdFlower code each comment into one of three classes: obscene, offensive, or clean. The inter-annotator agreement score is 87%. A list of unigrams and bigrams is created for the words and phrases that appear more than nine times in the dataset from the obscene and the offensive classes. This list is computed based on the Log Odds Ratio (LOR). The final list, after manual assessment, consists of a total of 288 words and phrases in addition to 127 hashtags.

In their 2018 paper on religious hate speech detection, Albadi, Kurdi, and Mishra (2018) create the first religious hate speech corpus in the Arabic language from Twitter data. They collect their dataset using Arabic keywords, covering the six main religions and beliefs in the Middle East: Islam, Sunni, Shia, Christianity, Judaism, and Atheism. The training dataset consists of 6,000 tweets, including 1,000 tweets representing each religion or belief, and the testing dataset consisting of 600 tweets, including 100 tweets representing each religion or belief. None of the datasets include retweets and reply tweets. CrowdFlower workers who satisfied predefined quality requirements annotate

both datasets. The quality requirements include Arabic speakers with IP addresses from Arabic countries, an assessment for the definition of religious hate speech with some examples, training on the annotation procedures, and a set of testing questions to measure each annotator's performance accuracy. Each tweet is annotated using a two-level hierarchy of labelling. Firstly, annotators code tweets into one of three classes: hate, not hate, or unclear and unrelated (later removed). Then, for the hate class, they further label its tweets into one of seven classes: Muslims, Jews, Christians, Atheists, Sunnis, Shia, and/or other. Inter-annotator agreement is computed to ensure agreement among coders. Moreover, they use the samples labelled as hate to generate three hate lexicons, using three well-known feature selection methods: chi-square statistical test for AraHate-Chi, Pointwise Mutual Information (PMI) for AraHate-PMI, and Bi-Normal Separation (BNS) sentiment scoring method for AraHate-BNS. All lexicons provide a hate score for each term.

Another publicly available Arabic hate speech and abusive language dataset is the Levantine Twitter Dataset for Hate Speech and Abusive Language (L-HSAB). The L-HSAB contains 5,846 tweets labeled as Hate (468 tweets), Abusive (1,728 tweets), or Normal (3,650 tweets) (Mulki et al., 2019). It is the first dialectal hate speech Arabic dataset. The overall theme of the dataset is politics. The authors used a set of keywords as the starting point to collect their dataset from the Twitter API from the time frame between March 2018 and February 2019. These keywords include, but are not limited to, "اللاجئين/refugees", "البنات/females", "العرب/Arabs", and "الدروز/Druze". They filter the initial dataset manually to ensure the collection of only Syrian and Lebanese tweets without

duplication. Then, they clean tweets to remove some characters (such as @, RT, and #) before presenting them to the annotators. Three native Levantine speaker annotators label the data. To avoid any personal bias or ambiguity, they were provided with clear definitions for each label, including a list of nicknames for political groups and races. Next, they use the dataset to create two lexicon lists, a hate speech words list, and an abusive words list. The lexicon lists provide a score associated with each word to indicate the degree of each word's relatedness to the class, hate or abusive. These degree scores are calculated using two words relationship techniques, word distribution for each class, and word-class correlation. The Cohen's kappa score among all raters exceeds 59%, and the inter-annotator agreement is 76.5%.

Similar to the L-HSAB dataset, the Tunisian Hate and Abusive speech (T-HSAB) dataset is provided by Haddad et al. (2019). The Tunisian dialect, also known as Derja, is a unique Arabic dialect as it has some words and phrases from other languages such as Amazigh, French, Turkish, and Italian. The main topic of T-HSAB is related to politics, social causes, religion, women's rights, and immigration. The extraction criteria depend on several phrases that are likely to be used in hate speech and abusive content, for instance "اليهود/Jews", "الأفارقة/Africans", and "المساواة في الميراث/gender equality in inheritance". The dataset was extracted for the period from October 2018 to March 2019. The dataset's total size is 6,075 comments, with 3,834 comments labeled as normal, 1,127 comments labeled as abusive, and 1,078 comments labeled as hate. The authors clean the text to remove platform-specific symbols; RT, user mention (@), and hashtags (#). The text was also preprocessed by removing emoji, digits, and all non-Arabic

characters. Three annotators were assigned for the annotation process. Various measures were used to measure the agreement among the annotators. The observed agreement, which is equal to the percentage of the agreed annotations out of the total number of annotations, is 81.82%. The Pairwise Percent Agreement between each annotator pair are 97.963%, 83.11% and 82.563%. Furthermore, the Cohen's kappa scores are 0.961, 0.638, and 0.624. The authors create two lists for the most frequently occured ten terms within the hate speech and abusive language categories. Each term is accompanied with percentage of distribution under the specific class. In addition to the distribution-based lists, two more lexicon lists were provided to check if the words assigned to each class of abusive language and hate speech were discriminatory. These two lists are created by calculating the word's correlation with the normal class, then assigning a hate score (HtS) and an abusive score (AbS) for the words that either mostly or rarely appeared in hate speech and abusive language classes.

The most commonly used dataset among surveyed literature is a dataset provided by the fourth workshop on Open-Source Arabic Corpora and Corpora Processing Tools (OSACT) in the Language Resources and Evaluation Conference (LREC) 2020 (Mubarak et al., 2020). Tweets were extracted based on two criteria: contained the vocative particle "يا/yA/O" and published between April 15 and May 6, 2019. The OSACT 2020 Arabic dataset is publicly available for free and consists of 10,000 tweets that are manually annotated by native Arabic speakers. Each tweet is labeled using a two-level labeling hierarchy. The top labeling level is binary: offensive or not offensive. Then, each tweet is further classified into hate speech or not hate speech. The dataset is

extremely imbalanced; out of the 10,000 tweets, only 1,900 are offensive, and out of these offensive tweets, only 95 are hate speech. Content of tweets is preprocessed to replace user mentions with @USER, URLs with URL, and empty lines with <LF>.

Aljarah et al. (2020) construct another Twitter hate speech dataset for the Arabic language, but it has fewer samples than the OSACT dataset. The final dataset delivered by Aljarah et al. (2020) contains of 3,696 tweets. Tweets are collected based on a list of phrases belonging to sport (e.g., "الوحدات/Alwahadat"— sports club), religion (e.g., "الإسلام/Islam"), racism (e.g., "العنصرية/racism") and journalism (e.g., "لاجئون/refugees"). Two annotators labeled the data using three classes: hate, not hate, or neutral. After removing duplicates and irrelevant tweets, the distribution of the dataset was 843 hate speech tweets, 790 non-hate speech tweets, and 2,061 neutral tweets.

The first multi-platform dataset for Arabic hate speech detection is constructed by Omar, Mahmoud, and Abd El-Hafeez (2020). The social media platforms used in collecting this dataset are Facebook, Twitter, YouTube, and Instagram. For each platform, Arabic accounts referencing controversial issues (such as religious, political, and sports-related topics) are selected. The data collection process consists of using a web crawler to extract data from each page automatically, and then saving the data into a text file. After collecting the dataset, the following data filtering and cleaning steps are performed: removing non-Arabic characters, emoji, and URLs; posts with less than two words were deleted. After that, three native Arabic speakers were assigned to label the samples as either hate speech or not hate speech. This process continues until a balanced dataset of 20,000 samples is formed for Arabic hate speech.

A very recent study by Chowdhury et al. (2020) releases the first dialectal Arabic Multi-Platform Offensive Language Dataset (MPOLD). The dataset has over 100k comments from international news agencies collected from three social media platforms: Facebook (20%), Twitter (40%), and YouTube (40%). The total number of comments is 4,000, with 84.13% not offensive comments and 16.88% offensive comments.

Table 2 shows the list of lexicons related to offensive content for the Arabic language, which was constructed by the surveyed studies (Albadi et al., 2018; Mubarak et al., 2017; Mulki et al., 2019). Albadi, Kurdi, and Mishra (2018), Haddad et al. (2019), and Mulki et al. (2019) give a score to each word in the list, thereby providing a measurement for the degree of association of that word to the class. Thus, the scores represent the degree of association of the word to the religious hate speech in Albadi, Kurdi, and Mishra (2018) and to hate and abusive content in Mulki et al. (2019). Mubarak, Darwish, and Magdy (2017) do not provide scores to their vulgar, pornographic, and obscene speech-related lexicons.

Table 2 Publicly available lexicons for Arabic offensive language

Ref.	Lexicons Purpose	Source	Size
Albadi et al. (2018)	Offensive/vulgar, religious/political, and war/violence	Twitter	1523 words
Mubarak et al. (2017)	Vulgar, pornographic, and obscene speech	Twitter	288 words and 127 hashtags
Mulki et al. (2019)	Levantine Hate Speech and Abusive Language	Twitter	10 abusive words and 10 hate words
Haddad et al. (2019)	Tunisian Hate Speech and Abusive Language	Not available	10 abusive words and 10 hate words

1.2 Tools and Techniques

One of the oldest publicly available tools for analyzing Arabic text is the MADAMIRA tool kit. MADAMIRA supports POS tagging, tokenization, diacritization, lemmatization, Named Entity Recognition (NER), and base form extraction (e.g., gloss). Albadi, Kurdi, and Mishra (2018) use MADAMIRA 2.1 in their study to perform tokenization, while Abdelfatah, Terejanu, and Alhelbawy (2017) use the same tool for tokenization and base form extraction (also called gloss).

There are multiple stemmers for the Arabic language, such as the light Arabic stemmer ARLSTem[7], which does not depend on any dictionary in processing texts. Instead, it is a rule-based stemmer that uses some rules to stripe prefixes, suffixes, and

[7] https://www.nltk.org/_modules/nltk/stem/arlstem.html

infixes. ARLSTem has been used by Alakrot, Murray, and Nikolov (2018a) in their offensive language detection study. Another Arabic stemmer is ISRI Arabic Stemmer[8], a light Arabic stemmer similar to ARLSTem, but the latest version has some new features. The latest version of ISRI Arabic Stemmer includes 90 more stop words than the previous one and the pattern "تفاعيل". Besides, it does not normalize all Hamza due to the ambiguity created by this process, which might change the original root. The NLTK tool kit provides both ARLSTem and ISRI Arabic Stemmer. The ISRI Arabic Stemmer has been used by Albadi, Kurdi, and Mishra (2018) in their hate speech detection study. Furthermore, the WEKA[9] tool kit supports Arabic. It has special packages for analyzing Arabic text. Haidar, Chamoun, and Serhrouchni (2017) use version 3.9.1 of WEKA in their cyberbullying detection study.

For offensive language detection, analyzing the sentiment can be valuable. Haidar, Chamoun, and Serhrouchni (2017) include features for analyzing Twitter users' sentiment polarity to train a classifier. They applied SentiStrength[10] as it allows for flexibility of language. SentiStrength produces parallel weights of positive and negative sentiments for short text by assigning a range of 1 to 5 for positive texts (5 for mostly positive) and a range of –1 to –5 for negative texts (–5 for mostly negative). SentiStrength supports languages other than English because it depends on a predefined set of words, along with their predefined sentiment scores, in creating text weights that can be customized. Thus, researchers can define their own customized profane terms set

[8] https://www.nltk.org/_modules/nltk/stem/isri.html
[9] https://www.cs.waikato.ac.nz/ml/weka/
[10] http://sentistrength.wlv.ac.uk/

and provide them to SentiStrength, then use the output to construct the SentiStrength feature vector to feed the classification model as Haidar, Chamoun, and Serhrouchni (2017) do in their study. This feature of SentiStrength is crucial for our study domain as Arabic tweets are written in different dialects and include slang, along with obfuscated offensive and abusive words— issues that could be mitigated by customizing sentiment inputs.

AraVec[11] is a pre-trained word embedding open-source project developed specifically for the Arabic language (Soliman et al., 2017). It has multiple models for word embedding based on two techniques—skip-gram and CBOW—and is based on three corpora: Twitter, World Wide Web pages, and Arabic Wikipedia articles. AraVec word embeddings have been used for detecting offensive language in Arabic YouTube comments (Mohaouchane et al., 2019) and for detecting religious hate speech on Twitter (Albadi, Kurdi, & Mishra, 2018).

The Arabic pre-trained Mazajak word embeddings[12] are initially developed by using a large Twitter corpus of 250M tweets. Then, another compact version is developed by using a smaller corpus of 100M tweets. Each embedding vector has a dimensionality of 300. Text is preprocessed by removing URLs, diacritics, emoji, and punctuation. Similar to AraVec, Mazajak has two architectures: skip-gram and CBOW. Hasan et al. (2020) use Mazajak for both offensive language detection and hate speech detection and demonstrate its beneficial effects on the classifiers' performance. Abu Farha and Magdy (2020) and Alharbi and Lee (2020) report similar findings to Hasan et al. (2020) using

[11] https://github.com/bakrianoo/aravec
[12] http://mazajak.inf.ed.ac.uk:8000/

Mazajak to generate word embeddings boosts the performance of the classifier from others that do not depend on it.

Multilingual BERT (M-BERT)[13] is proposed by Google Research and has two versions. BERT stands for Bidirectional Encoder Representations from Transformers. It is an innovative language model that presents SOTA results in multiple NLP tasks, such as question answering and language inference (Devlin et al. 2018). The first multilingual BERT version is the BERT-base-multilingual-uncased model, which covers 102 languages, including Arabic. The second version is the BERT-base-multilingual-cased model, which covers 104 languages, including Arabic. Both models consist of 12 layers, 768 hiddens, 12 heads, and 110M parameters. Wikipedia dumps of each language (excluding user and talk pages) are used to train the models with a shared word piece vocabulary. For some low resource languages, the percentage of their Wikipedia content is tiny compared to other languages. Thus, an exponentially smoothed weighting technique is used to up-sample them before training the model. Text is preprocessed by lower casing, accent removal, punctuation splitting, whitespace tokenization, and for Chinese language, Japanese Kanji, and Korean Hanja text, spaces added around every character in the CJK Unicode range before applying WordPiece tokenization. M-BERT outperforms other tools when it applies to multilingual text; however, it shows some limitations in tokenizing Arabic sentences, which could degrade the classifier's performance (Eljundi et al., 2019). This finding is in line with other experiments conducted by Hasan et al. (2020), Saeed et al. (2020), and Keleg et al. (2020), which

[13] https://github.com/google-research/bert/blob/master/multilingual.md

report poor performance in Arabic offensive language and hate speech detection in comparison to other word embeddings, machine learning classifiers, and deep learning classifiers. In addition, Abu Farha and Magdy (2020) try M-BERT with the BertAdam optimizer, train the model with 4 epochs, a learning rate of 1e−5, and set the maximum sequence length to the maximum length seen in the training set. However, the results are not as good as those obtained from Bi-LSTM, CNN-BiLSTM, and multitask learning models. In (Elmadany et al., 2020), multiple M-BERT-based classifiers were used with different fine-tuning settings for offensive language and hate speech detection tasks, and in both tasks, the reported macro F1 score was not better than what has been reported by other studies using simple traditional machine learning methods.

Facebook AI researchers; Lample and Conneau; first introduced the XLM model (also called XLM-100) in January 2019. Wikipedia Data covering 100 languages are used in pre-training the XLM model. The vocabulary size of the model is 95K. Like M-BERT, it depends on the Transformer architecture, which applies the attention mechanism to learn contextual relationships among words or sub-words. The architecture includes 1024 hidden units, 8 heads, GELU activations, 0.1 dropout rate, Adam optimizer, linear warmup, and learned positional embeddings. The learning rate varies from 10^{-4} to 5.10^{-4} (Lample & Conneau, 2019).

In November 2019, the XLM-RoBERTa (also called XLM-R) scaled up the XLM model (Conneau et al., 2019). More than two terabytes of Wikipedia Data and filtered CommonCrawl data covering 100 languages are used in pertaining XLM-RoBERTa. The construction of XLM-RoBERTa is very similar to that of XLM.

While M-BERT and XLM-R support various languages, Arabic-specific BERT models have been used by some of the surveyed studies, such as AraBERT and Arabic-BERT (also called BERT-base-Arabic). AraBERT[14] is an Arabic version of the BERT model that uses BERT-base configuration and 12 encoder blocks with 768 hidden dimensions (Antoun, Baly, & Hajj, 2020). It also uses 12 attention heads, 512 maximum sequence length, and around 110M parameters. Multiple MSA corpora were used to train the model, which include: manually scraped Arabic news websites for articles, 1.5 billion words extracted from news articles from ten primary news sources, and OSIAN, which is an Open-Source International Arabic News Corpus (Antoun, Baly, & Hajj, 2020). The authors apply some sub-word unit segmentation techniques to reduce unnecessary redundancy by using the segmentation tool, Farasa, to break words into stems, prefixes, and suffixes. After that, the segmented pre-trained dataset was trained to create the vocabulary using the language independent tokenizer, SentencePiece. The same tokenizer is also used on the pre-training dataset without performing sub-word unit segmentation to evaluate its effects, creating a second version of AraBERT called AraBERTv0.1. (Djandji et al., 2020) adopts a multitask learning approach with the AraBERT model for offensive language detection and hate speech detection and reports its superiority over single-task and multilabel approaches. Keleg, El-Beltagy, and Khalil (2020) used both M-BERT and AraBERT with the same experiment settings and reported an increase in macro F1 score by 5% with AraBERT.

[14] https://github.com/aub-mind/arabert

Arabic-BERT model is another Arabic monolingual BERT model. Safaya et al. (2020) develop the Arabic-BERT model using around 8.2 billion words equivalent to 95GB of text. The pre-training corpus consists of multiple Arabic resources such as Arabic OSCAR and Arabic Wikipedia. It includes MSA and dialectal Arabic, in addition to some English words which were kept because of their significance on NER. The repository of Google BERT is used to train the model with a single TPU v3-8 from TensorFlow Research Cloud (TFRC). The authors make some changes to the original training settings of BERT, they apply 3M training steps with 128 batch size, instead of 1M with 256 batch size. Results on sentiment analysis show a higher F1 score for Arabic-BERT over M-BERT and hULMonA (SOTA model for Arabic sentiment analysis) when used with Levantine dialect and Egyptian dialect datasets.

AraNet (Abdul Mageed et al., 2020) is another Arabic language tool that has been used for the task of offensive language detection. AraNet is a deep learning tool for Arabic social media processing. It can predict the age, dialect, gender, emotion, irony, and sentiment of the text. This tool aims to free the prediction model from feature engineering, which has been implemented by using the M-BERT model in developing the tool. AraNet is used by Elmadany et al. (2020) to predict the sentiment of tweets. The purpose of this study was to augment the primary dataset with an external set of unlabeled tweets to increase the offensive language and hate speech samples. In (Elmadany et al., 2020), the authors hypothesize that tweets with negative sentiment are more likely to have offensive content, thus, they use them to create a more balanced dataset.

FastText[15] is an open-source library for text classification and representation learning. The library has pre-trained models for 157 different languages, including Arabic. FastText applies preprocessing strategies to support position dependency, phrase representations, and use of sub-word information. To support position dependency, a position-dependent weighting representation was implemented by learning position representations and applying them to re-weight the word vectors. In addition, instead of using unigram as in CBOW and skip-gram, FastText uses word n-grams to capture richer information. It uses the Word2Phrase tool from the Word2Vec software library to perform the task of phrase representation. In (Hassan et al., 2020) and (Saeed, Calders, & Kamiran, 2020), FastText is outperformed by other classifiers in general offensive language classification and hate speech classification. Alharbi and Lee (2020) demonstrate in their study the importance of incorporating character-based embeddings that FastText obtains with other types of word embeddings to increase the performance of the classifier for offensive language and hate speech tasks.

2. Approach

I group the studies based on the similarity of the targeted offensive language tasks tackled in the studies. Studies fall into five main categories shown in Table 3.

[15] https://fasttext.cc/

Table 3 Examples of each offensive category

Categories	Examples
General offensive or abusive	Abusive users, abusive posts, profanity
Hate speech	Religion, sport, racism, politics
Cyberbullying	Threatening posts, harassment, flaming
Adult content	Pornography, nudity
Violence	Extremism, terrorism

1.1 General Offensive (Abusive) Language

This category includes studies that do not focus on a particular type of offensive language. Instead, they study offensive (abusive) content on social media using a more inclusively.

Previous studies focus on detecting offensive Arabic tweets to identify abusive Twitter accounts (Abozinadah, Mbaziira, & Jones, 2015; Abozinadah, 2017; Abozinadah & Jones, 2017). Abozinadah, Mbaziira, and Jones (2015) build an initial dataset from 500 Twitter accounts based on a set of Arabic swear words. Then, they check the 50 most recent tweets, profile pictures, and hashtags for each of these 500 Twitter accounts to extend their dataset. They extend the initial dataset until they reach a dataset of 350,000 Twitter accounts and 1,300,000 tweets. The core dataset has balanced classes; half of the samples are labeled abusive, and the other half are labeled non-abusive. The text was pre-processed by removing all non-Arabic words, symbols, numbers, and stop words, normalizing Alif, Alif Maqsura, and Ta Marbouta, and correcting misspelled words.

Then, three types of features are used to feed the classifiers, including profile-based features, tweet-based features, and social graph features. Profile-based features apply some statistical measurements on the number of tweets, followers, and following for each user account. Tweet-based features, using the most recent tweets for each account, contain statistical measurements and NLP techniques (e.g., BOW) on text, hashtags, and mentions, in addition to analyzing website's links within tweets using VirusTotal. Social graph features use social graph theory concepts, including eigenvector, out-degree, and in-degree for each account. They develop three classifiers: NB, SVM, and Decision Tree, and explore them using multiple features and datasets. Results show that NB outperforms the other classifiers when used with 100 features and ten tweets for each account with an accuracy score of 85%. Findings show some patterns for abusive accounts, such as using more hashtags than non-abusive accounts, with hashtags that cover various unrelated topics such as names of cities, countries, profanity, or slang words (Abozinadah, Mbaziira, & Jones, 2015; Abozinadah, 2017).

Alakrot, Murray, and Nikolov (2018a, 2018b) have also studied automatic Arabic offensive language detection. They created a dataset from YouTube comments based on selecting channels that have controversial videos about celebrities. Their final dataset includes 167,549 comments posted by 84,354 users and 87,388 replies posted by 24,039 users from 150 YouTube videos (Alakrot, Murray, and Nikolov, 2018a). Three annotators from different Arabic nationalities, representing the same nationalities of the posters' comments, labeled each comment, and then the inter-annotator agreement is computed. Two labels are used: positive for offensive comments and negative for not

offensive ones, while unclear comments are kept unlabeled (Alakrot, Murray, and Nikolov, 2018a). This dataset is then split into training and testing datasets and used to train an SVM-based classifier. Multiple pre-processing steps and features were explored to arrive at the best performance model. The light Arabic stemmer ARLSTem was used to convert text to tokenized stems of words (Alakrot, Murray, and Nikolov, 2018b). Moreover, comments were filtered by removing words matched with the stop words list provided by the NLTK and removing non-alphabetic characters (e.g., punctuation, numbers), diacritics, and Kashidas, which is a justification used in cursive scripts to elongate characters rather than separate them by spaces. Text pre-processing includes text normalization in which some letters are replaced by other letters, such as Alif (أ/إ/آ/ا) to Alif Maqsura (ى/ي) and Ta Marbouta (ة) to (ه). Pre-processing also includes replacing Persian and Urdu letters with equivalent Arabic letters and correcting commonly mistaken letters that have phonetic similarity (e.g., Dhaah (ظ), Dza'a (ض)). Two features were explored: n-gram (n = 1-5) and word-level. Best performance scores were achieved using the SVM-based classifier with 10-fold cross-validation and word-level features after fully pre-processing texts, which shows 90.05% accuracy (Alakrot, Murray, & Nikolov, 2018b).

Mohaouchane, Mourhir, and Nikolov (2019) study multiple deep learning models to classify offensive Arabic language for YouTube comments using the same dataset developed by Alakrot, Murray, and Nikolov (2018a). They apply the same pre-processing procedures of Alakrot, Murray, and Nikolov (2018b) discussed earlier. They develop word embeddings using AraVec trained on the Twitter dataset and skip-gram model.

Four deep learning models were evaluated for the purpose of classifying offensive comments, including CNN, Bi-LSTM, Bi-LSTM with attention mechanism, and combined CNN and LSTM. For the CNN classifier, they specify the length of the comment as 418 words, thus, comments less than 418 words are padded with 0s. For hyperparameter tuning, they use the Tree-Structured Parzen Estimator (TPE) algorithm for Bayesian hyperparameter optimization. Results demonstrate an overall better performance for CNN with the highest accuracy of 87.84%, precision of 86.10%, and F1 score 84.05%, while the combined CNN-LSTM model shows a better recall score of 83.46%. Their findings can best be summarized with the following main points: (1) convolutional layers improve the performance of the models; (2) more offensive comments are detected, and better recall is obtained when combining convolutional layers with LSTM; and (3) the Bi-LSTM model with the attention mechanism improves the recall score of the model, detecting more offensive comments.

An analysis for the Tunisian dialect is conducted by Haddad et al. (2019) who uses the T-HSAB dataset previously discussed to detect abusive language. The study deployed an SVM- and an NB-based system to predict whether the sample text includes abusive language. The authors explore multiple n-gram sizes with term frequency weighting to arrive at the best feature size. The results reveal better performance by the NB-based model than the SVM-based model when used with a combination of unigram and bigram features. The NB model reports 92.3% in F1, 91.5% in recall, 93.5% in precision, and 92.9% in accuracy, whereas the SVM-based model reports 74.7% in F1 score, 73.8% in recall, 76.4% in precision, and 77.7% in accuracy.

Many studies use the OSACT dataset. In this section, I consider the experiments that cover the first labeling hierarchy because it identifies whether a tweet contains any offensive language or not. In the next section, I discuss the experiments that focus on the second labeling hierarchy for hate speech detection. For OSACT Arabic offensive language detection, the highest achieved macro F1 score is 90.5%, recorded by Hassan et al. (2019). In (Hassan et al., 2019), the authors applied four pre-processing steps: (1) removing all words that consist of non-Arabic characters; (2) removing all diacritics; (3) removing all punctuation; and (4) replacing repeated characters with only one. Multiple machine learning and deep learning classifiers are studied, but the authors only record results obtained by the best-performing classifiers. The best model use an ensemble of an SVM classifier with 1 to 5 n-gram characters and 1 to 3 n-grams word features combined with pre-trained Mazajak word embeddings, a Feed Forward Neural Network (FFNN) classifier with 1 to 8 n-gram characters, and 1 to 4 n-grams word features, and a CNN classifier with pre-trained Mazajak word embeddings. This ensemble classifier records 90.5% for the macro F1 score, 93.9% for the accuracy, 90.2% for the precision, and 90.9% for the recall. Hassan et al. (2020) also developed other classifiers, including FastText (sub-word), CNN-BiLSTM, and M-BERT.

Djandji et al. (2020) use AraBERT for the OSACT dataset with a multitask learning approach and a multilabel classification approach. Multitask learning solves the data imbalance problem in the OSACT dataset by leveraging information from multiple tasks simultaneously. The same study also uses a multilabel classification approach using AraBERT, in which all labels of the two labeling hierarchies in the OSACT dataset—

offensive and hate—are merged under a broad task of violence detection. Both approaches are explored as attempts to account for the extreme imbalance of the dataset. The authors apply pre-processing steps similar to the ones used in creating AraBERT to ensure its consistency. The pre-processing steps are: (1) using the Farasa Arabic segmenter to tokenize words; (2) removing users' mentions, retweet mentions, URL, diacritics, emoji, and newline tokens; and (3) replacing underscores in hashtags with white spaces and padding the hashtags with white spaces. All experiments are conducted with 5 epochs, 32 batch size, and 256 sequence length in a GPU-accelerated environment. Results report 90.15% as the highest macro F1 score for offensive language detection when adopting the multitask learning approach with AraBERT. Findings from Djandji et al. (2020) reveal the superiority of using a multitask learning approach over a multilabel classification approach when using AraBERT for offensive language detection.

Keleg, El-Beltagy, and Khalil (2020) investigate several classification models—AraBERT, M-BERT, CNN, Bi-LSTM, and LR—for offensive language detection using the OSACT dataset. Different pre-processing procedures are applied for each classifier, removing diacritics, fixing elongated words, and tokenization. Many features are also developed, including 1 to 9 character n-grams TF-IDF, AraVec word embeddings, and Word2Vec. The AraBERT model augmented with a list of profane words records the highest macro F1 score of 88%, followed by the Bi-LSTM model with 86.9%, the CNN model with 86.7%, the M-BERT with 82.6%, and the LR model with 74.6%. During the error analysis, Keleg, El-Beltagy, and Khalil (2020) highlight some issues that confuse

the classifier, such as the presence of offensive words in a non-offensive context, using sarcastic speech to quote scenes from popular movies, and wrong annotations.

The OSACT dataset has been studied using multitask learning and deep learning models by Abu Farha and Magdy (2020). The Arabic word embeddings Mazajak are used to create the word vectors; thus, the authors use the same pre-processing procedures that were used in creating Mazajak to the tweets. These pre-processing procedures include letter normalization for Hamza, Ya, and Ha; removing elongation; and basic cleanings, such as removing unknown characters, diacritics, punctuation, and URLs. Abu Farha and Magdy (2020) investigate various classifiers: Multinomial Naive Bayes (MNB), M-BERT, Bi-LSTM, and CNN-BiLSTM. In addition to a multitask learning classifier that consists of a CNN-BiLSTM trained on offensive language, hate speech, and sentiment, this multitask learning classifier had three versions: the first model trained just on offensive language and hate speech without sentiment; the second one trained on offensive language, hate speech, and sentiment predicted by Mazajak sentiment analyzer; and the third model trained on offensive language, hate speech, and sentiment, which assumes all offensive and hate speech tweets are negative sentiment and others were positive sentiment. The results report 90.4% as the highest macro F1 score in offensive language detection, recorded by the third multitask learning model.

In (Saeed, Calders, & Kamiran, 2020), the authors created a mixture of multiple deep learning models and an ensemble classifier of traditional machine learning models to the OSACT dataset in building an offensive language detection system. Tweets are pre-processed by removing replaced tokens for user mentions, URL, and newline;

removing emoticons, emoji, punctuation, English characters, digits, and diacritics; character normalization for Hamza, Ya, Ha, and Qaf; removing character repetitions of more than three; and special pre-processing for Out-Of-Word-Embeddings-Vocabulary (OOWEV) words, in which the OOWEV words are split into two tokens whenever the first character of the word is Wa, Fa, or Sa. The last pre-processing step is included because Wa, Fa, or Sa are often used at the beginning of the words as a grammatical particle (Wa as "and" or "vow" or "oath", Fa to show consequence to a former statement, and Sa as a reference for a very close future). Saeed, Calders, and Kamiran (2020) apply a concatenation of two word embeddings to form the feature vectors. Each vector contains a 300-dimensional FastText vector and a 100-dimensional Word2Vec vector, which result in a 400-dimensional vector representation. The system architecture has two main classification phases. The first classification phase consists of four deep learning models: CNN, Bi-LSTM, Bidirectional Gated Recurrent Unit (Bi-GRU), and CNN-BiLSTM. The predicted results of the first phase are used as the new training set for the second phase, which consists of a stacking ensemble classifier of an NB, a LR, an SVM, a Nearest Neighbour, and a Random Forest. The system records a macro F1 score of 95.51% for offensive language detection.

Alharbi and Lee (2020) create a system for offensive language detection that used the OSACT dataset. In (Alharbi & Lee, 2020), the following steps are performed during the pre-processing phase: removing non-Arabic characters, diacritics, and punctuation; letter normalization for Hamza and Ta Marbouta; and specific segmentation for strings that start with "يا/yA/O" that use RegEx to segment the string into two words. Moreover,

emojis are kept during the preprocessing phase and later each emoji is used to create a vector representation. The model is based on multiple word embedding tools: AraVec and Mazajak for word level embeddings and FastText for character level embeddings. The authors emphasize the significance of having the two levels of embeddings because the word level embeddings provide valuable insights into the semantic similarity and the character level embeddings support the encoding of all variants of a word's morphology. Two classifiers are developed; the first one that generates the highest macro F1 score is the LSTM deep learning model, and the second one is the tree decision-based algorithm "XGBoost" model. The LSTM shows 88.5% in the macro F1 score, 93.5% in the accuracy, 87.1% in the recall, and 90.1% in the precision. The XGBoost shows 87% in the macro F1 score, 92.8% in the accuracy, 84.9% in the recall, and 89.5% in the precision in offensive language detection.

In Haddad et al. (2020), the OSACT dataset has also been used in training and developing an offensive language detection system. The following steps are taken during the pre-processing phase: (1) removal of non-Arabic words, diacritization, punctuation, emoticons, and some stopwords; (2) replacing some words with their Arabic equivalents, such as "URL" with "يورل"; reducing elongated words and consecutive repetitive characters to their original form; and (3) letter normalization for Alif Maqsura, and Ta Marbouta. After that, Haddad et al. (2020) try to solve the issue of imbalanced classes between offensive and not offensive samples by adding more offensive samples from the YouTube dataset provided by Alakrot, Murray, and Nikolov (2018a). The AraVec word embeddings are used to create word representations. The authors check the overlap

between tokens from the augmented dataset for offensive language and the corpus used in developing the AraVec word embeddings, which shows 83% overlap. The baseline models —SVM, LR, and Ridge—TF-IDF and BOW vectorizers features are studied. Moreover, several deep learning classifiers are explored, including CNN, CNN with attention, GRU, Bi-GRU, and Bi-GRU with attention. The best performing model is the Bi-GRU with attention, which reports 85% in the macro F1 score, 91% in the accuracy, 88% in the precision, and 83% in the recall. The authors recommend advancing the system by using LSTM instead of GRU to help in capturing long range dependencies in samples.

In another OSACT dataset-based study, Elmadany et al. (2020) utilize a list of 352 manually constructed offensive words to develop an offensive language detection system based on assuming a correlation between negative sentiment and offensive language. The initial list of words includes 2,158 words extracted from the training OSACT dataset based on one criterion that consists of appearing after the "يا/yA/O". Then, the initial list is manually investigated to construct the final offensive lexicon list. After that, the lexicon lists are used as keywords to extract tweets from another Twitter corpus constructed by Abdul Mageed et al. (2019). This list of the extracted tweets was further studied to check their sentiments by applying AraNet (Abdul Mageed et al., 2020). Only the negative tweets are kept in order to augment the OSACT dataset as an attempt to solve the imbalanced distribution of the classes. The M-BERT model forms the base for generating all classifiers used in this study. Four classifiers are created, including M-BERT fine-tuned on the OSACT training dataset, M-BERT fine-tuned on

the Arabic binary sentiment analysis corpus provided by Abdul Mageed et al. (2019), M-BERT fine-tuned on eight classes of the Arabic emotion identification corpus provided by Abdul Mageed et al. (2019), and the last model is similar to the previous one but the M-BERT is fine-tuned on both the augmented dataset and the multi-classes emotion identification corpus. The last model that utilizes the augmented dataset in fine-tuning the M-BERT model achieves the best performance; its macro F1 score is 82.85%, and the accuracy is 89.35%. The authors end the study with a recommendation to further analyze semi-supervised methods to improve the offensive language classification model for the Arabic language.

The only offensive language detection system that used a multi-platform dataset is created by Chowdhury et al. (2020). In this study, the authors use many datasets to evaluate the system, including the MPOLD (Chowdhury et al., 2020), the L-HSAB (Mulki et al., 2019), the Aljazeera.net deleted comments datasets, and the Egyptian tweets dataset developed by Mubarak, Darwish, and Magdy (2017). Text is pre-proceeded by tokenization, removing punctuation, URLs, stop words, and diacritics, and adding whitespace to separate adjacent words and emoji. Moreover, all emojis and hashtags found in the text are kept because of their contextual information. Then, the text is converted to 1 to 5-character TF-IDF vectors. Experiments are designed to test for the model generality; thus, the authors use an SVM-based model and assess it using platform-wise 5-fold cross-validation by following an approach called Leave-One-Platform-Dataset-Out (LOPO). For instance, a model trained on a Twitter dataset is tested using a Facebook dataset, an Aljazeera.net dataset, a YouTube dataset, and different

combinations of them. Findings show low generalizability and poor performance by the model trained using one platform dataset compared to those that are trained using multiple-platform datasets. The highest reported macro F1 score is 84%, which is achieved by the model trained using a combination of Facebook, YouTube, and Aljazeera.net datasets and tested on a Twitter dataset. Moreover, this study also investigates the performance of the classifiers on dialectal Arabic—Levantine and Egyptian. The model trained on the multi-platform Aljazeera.net comment dataset model outperformed the other models in capturing diversity in different Arabic dialects. Moreover, the same study conducts an analysis of the use of emojis within offensive samples, which indicates more frequent usage of emojis from the animal category compared to the other categories of emoji. For instance, the model trained on both the Twitter and the YouTube samples performs significantly better in detecting offensive content on the Facebook test set than the individual YouTube or Twitter models.

Table 4 summarizes the previous studies for offensive and abusive language detection we discussed above.

Table 4 Qualitative analysis of general offensive or Abusive language detection literature

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Abozinadah et al. (2015)	Removing non-Arabic words, symbols, numbers, and stop words, normalizing letters, correct, misspelling	Profile-based features, tweet-based features, and social graph features	Twitter, 350,000, Twitter account, and 1,300,000 tweets	NB, SVM and Decision Tree	85% NB, 64% SVM, 84% Decision Tree	85% NB, 63% SVM, 84% Decision Tree	85% NB, 64% SVM, 84% Decision Tree	85% NB, 67% SVM, 84% Decision Tree
Alakrot et al.	Stemming, removing stop	Word-level	YouTube, 15,050	SVM	90.05%	82%	77%	88%

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
(2018b)	words, and non-alphabetic characters, normalizing letters, replacing Persian Urdu letters with equivalent Arabic letters, correct misspellings	features	comments					
Mohaouchane et al. (2019)	Adopt Alakrot, Murray, and Nikolov (2018b) approach	AraVec trained on Twitter dataset and skip-gram model	YouTube, 15,050 comments	CNN, Bi-LSTM, Bi-LSTM with attention mechanism, and combined CNN-LSTM	87.84% CNN, 86.42% Bi-LSTM, 85.75% Bi-LSTM with attention, 87.27% combined CNN-LSTM	84.05% CNN, 82.33% Bi-LSTM, 81.70% Bi-LSTM with attention, 83.46% combined CNN-LSTM	82.24% CNN, 80.97% Bi-LSTM, 81.51% Bi-LSTM with attention, 83.46% combined CNN-LSTM	86.10% CNN, 83.74% Bi-LSTM, 85.75% Bi-LSTM with attention, 83.89% combined CNN-LSTM
Haddad et al. (2019)	Removing non-Arabic characters, Emoji, numbers, RT, usermention, and hashtags	N-grams and term weighting features	N/A, 6,075 comments	NB and SVM	92.9% NB, 77.7% SVM	92.3% NB, 74.7% SVM	93.5% NB, 73.8% SVM	93.5% NB, 76.4% SVM
Hassan et al. (2020)	Removing non-Arabic characters, diacritics, punctuation, and replacing repeated characters with one	N-grams and Mazajak word embeddings	Twitter, 10,000 tweets	SVM, CNN, CNN-BiLSTM, FFNN, FastText, M-BERT, and ensemble	94.3% SVM, 92.1% CNN, 92.9% CNN-BiLSTM, 91.9% FFNN, 89.5% FastText 90.1% M-BERT 94.3% ensemble	90.4% SVM, 86.5% CNN, 87.8% CNN-BiLSTM, 85.1% FFNN, 79.4% FastText 83.3% M-BERT 90.5% ensemble	91% SVM, 86.7% CNN, 88.1% CNN-BiLSTM, 84% FFNN, 75.7% FastText 83.7% M-BERT 91.1% ensemble	89.8% SVM, 86.2% CNN, 87.6% CNN-BiLSTM, 87.1% FFNN, 85.5% FastText 83% M-BERT 89.9% ensemble
Djandji et al. (2020)	Tokenization, removing usersmention, retweet, URL, diacritics emojis and newline, replacing underscore in hashtag with white spaces	Not available	Twitter, 10,000 tweets	AraBERT	Not available	90.15%	Not available	Not available

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Husain (2020b)	Converting emoji and emoticon to text, letter normalization, replacing repeated letter with one, dialect normalization, converting hyponym to hypernym, replacing underscore in hashtag with white space, removing numbers, HTML tags, more than one space, symbols, stopwords, and diacritics	Count vectorizer	Twitter, 10,000 tweets	SVM	90.2%	89.8%	90.2%	89.9%
Husain (2020a)	Adopt Husain (2020b) approach	Count vectorizer and TF-IDF	Twitter, 10,000 tweets	SVM, LR, Decision Tree, Bagging, Random Forest, and Adaboost	Not available	82% SVM, 81% LR, 69% Decision Tree, 88% Bagging, 87% Random Forest, and 86% Adaboost	Not available	Not available
Keleg et al. (2020)	Removing diacritics, fixing elongated words, and tokenization	N-grams, TF-IDF, AraVec word embeddings, and Word2Vec	Twitter, 10,000 tweets	LR, CNN, BiLSTM, M-BERT, and AraBERT	88.8% LR, 92.8% CNN, 92% BiLSTM, 90.5% M-BERT 92.8% AraBERT	74.6% LR, 86.7% CNN, 86.9% BiLSTM, 82.6% M-BERT 87.6% AraBERT	69.4% LR, 83.8% CNN, 88.4% BiLSTM, 80.5% M-BERT 87.1% AraBERT	92.1% LR, 90.6% CNN, 85.6% BiLSTM, 85.5% M-BERT 88.1% AraBERT
Abu Farha and Magdy (2020)	Normalizing letters, removing elongation, unknown characters, diacritics, punctuation, and URLs	N/A	Twitter, 10,000 tweets	NB, M-BERT, BiLSTM, CNN-BiLSTM, and multitask	N/A	49% NB, 85.7% M-BERT, 89.6% BiLSTM, 90.1% CNN-BiLSTM, and 90.4% multitask	N/A	N/A
Saeed et al.	Normalizing	FastText	Twitter,	NB, LR,	Not	64.73%	Not	Not

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
(2020)	letters, removing usermention, newline, emoticons, emojis diacritics, English characters, punctuation, and URLs, and process out-of-word-embeddings-vocabulary	and Word2Vec	10,000 tweets	SVM, Random Forest CNN, BiLSTM, BiGRU, CNN-BiLSTM, and ensemble	available	NB, 84.55% LR, 84.86% SVM, 80.92% Random Forest 88.67% CNN, 89.02% BiLSTM, 88.75% BiGRU, 87.84% CNN-BiLSTM, and 95.51% ensemble	available	available
Alharbi and Lee (2020)	Removing diacritics, punctuation, and non-Arabic characters, normalizing letters, and specific segmentation for يا/yA/O words	Mazajak, AraVec, and FastText word embeddings	Twitter, 10,000 tweets	LSTM and XGBoost	93.5% LSTM, 92.8% XGBoost	88.5% LSTM, 87% XGBoost	87.1% LSTM, 84.9% XGBoost	90.1% LSTM, 89.5% XGBoost
Haddad et al. (2020)	Removing diacritics, punctuation, non-Arabic characters, emoticons, and stopwords, replacing some tokens by their Arabic words, normalizing letters, reducing repeated letters and elongated words	AraVec word embeddings	Twitter, 10,000 tweets and YouTube, N/A\	SVM, LR, Ridge, CNN, GRU, and BiGRU	90.6% SVM, 87.2% LR, 90.5% Ridge, 92% CNN, and 91% BiGRU	83% SVM, 78% LR, 83% Ridge, 86% CNN, and 85% BiGRU	81% SVM, 79% LR, 80% Ridge, 85% CNN, and 86% BiGRU	85% SVM, 77% LR, 85% Ridge, 86% CNN, and 88% BiGRU
Elmadany et al. (2020)	Not available	Not available	Twitter, 10,000 tweets	M-BERT	89.35%	82.85%	Not available	Not available
Chowdhury et al. (2020)	Removing diacritics, punctuation, stop words, and URLs,	TF-IDF	Facebook, 800 posts, Twitter, 1,600 tweets,and	SVM	Not available	84%	Not available	Not available

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
	and adding a whitespace to separate between adjacent words and emoji		YouTube, 1,600 comments					

2.2 Hate Speech

Albadi, Kurdi, and Mishra (2018; 2019a) study religious hate speech in the Arabic language from Twitter. As part of their study, they collect their dataset using Arabic keywords representing multiple religious beliefs, as we discussed earlier. Prep-rocessing techniques used include normalizing some letters (e.g., Alif, Alif Maqsura, and Ta Marbouta) and removing diacritics (Albadi, Kurdi, & Mishra, 2018). SRIStemmer and MADAMIRA 2.1 were used to support the pre-processing tasks of stemming and filtering texts. They used the lexicons AraHate-PMI, AraHate-Chi, and AraHate-BNS as the basis for their baseline model, giving a score to each tweet based on matches of its content to the lexicon terms. They also develop an LR-based model and an SVM based model with a character n-gram feature (n = 1 to 4), and a GRU based RNN network with the Twitter-CBOW 300-dimension embedding model provided by AraVec, batches of size 32, and Adam as the optimizer (Albadi, Kurdi, & Mishra, 2018). Findings reveal that for some religious minorities in the Middle East—Jews, Atheists, and Shia—almost half of the tweets which mention them do so within the context of hate speech. Results record the best performance score of 77% F1 when using the GRU-based model. Training the same model with the additional temporal, user, and content features results in a SOTA

recall score of 84% (Albadi, Kurdi, & Mishra, 2019a). They extend their investigation of Arabic religious hate speech on Twitter to study its impact on malicious bots that are often used to amplify fake news and misinformation (Albadi, Kurdi, & Mishra, 2019b). Using the same dataset they developed earlier; they were able to define a set of 450 manually-labeled accounts with bot-like behaviors.

Chowdhury et al. (2019) extend the previously discussed studies by Albadi, Kurdi, and Mishra (2018; 2019a; 2019b) to study the effects of community interaction and social representations in detecting religious hate speech in the Arabic Twitter-sphere. They use the same dataset constructed by Albadi, Kurdi, and Mishra (2018), but due to the unavailability of some samples, the size of the dataset is reduced to 1,685 hate tweets and 2,265 not hate tweets. This study assumes that similar people are more likely to form online communities, in which similarity can be defined based on several factors, such as location, age, language, etc. Hence, if some information about a member of an online community is available, one can infer information about other individuals in the same community who are identified as similar according to a predefined similarity measure. Chowdhury et al. (2019) define this concept as leveraging community interaction. Accordingly, if an individual commonly shares hate speech content, then it is more likely that other individuals within the same online community are posting similar content. They adopt the same pre-processing approach of Albadi, Kurdi, and Mishra (2018), in addition to some other Twitter-specific pre-processing techniques, such as normalizing links, user mentions, and numbers to somelink, someuser, and somenumber. Two social network graphs were employed, a follower graph and a retweet graph, both of which

were unweighted, undirected graphs. These graphs were used to create a vector representation for each author using the Node2Vec framework to ensure having a node representation that gives equal weight to both structural and community-oriented information. In addition, they develop multiple features, including word embedding, node embedding, sentence representation, and character n-gram features (n = 1-4). Several classification models were explored using multiple combinations of features, such as GRU, LR, SVM, LSTM, Bi-LSTM, CNN, Bi-GRU, in addition to combining multiple models and using self-attention and Node2Vec criteria. The highest accuracy score recorded in the results is 81%, obtained by the model that contains a combination of Bi-GRU, CNN, and Node2Vec. Meanwhile, the best F1 score, recall, and precision were registered by the model that combines the LSTM, the CNN, and the Node2Vec; 89%, 78%, and 86%, respectively.

In Haddad, Mulki, and Oueslati (2019), the authors create the T-HSAB dataset and develop a system that could differentiate among hate speech, abusive language, and normal social media content in the Tunisian dialect. Two traditional machine learning classifiers are created—SVM and NB—which are trained with uni-grams, uni-grams and bi-grams, and uni-grams and bi-grams and tri-grams, then, term frequency weighting is applied to reduce the dimensionality of the features. The SVM model reports 62.2% for the F1 score, 59.9% for the recall, 66.5% for the precision, and 73.9% for the accuracy. On the other side, the NB model outperforms the SVM in all performance measures. The NB model achieves 83.6% in the F1 score, 79.8% in the recall, 89.5% in the precision, and 87.9% in the accuracy.

Omar, Mahmoud, and Abd El-Hafeez (2020) release the first multi-platform dataset for Arabic hate speech detection, and they also use it to create a binary system for hate speech detection based on 12 machine learning classifiers and two deep learning classifiers. Multiple machine learning models are explored, including MNB, Complement NB, Bernoulli NB, SVM, NuSVC, LinearSVC, LR, Decision Tree, Stochastic Gradient Descent (SGD), Ridge, Perceptron, and Nearest Centroid, in addition to the CNN and the RNN deep learning models. The best F1 score is 98.7% recorded by the RNN model, which also reports the same percentage in recall, precision, and accuracy.

The Twitter hate speech dataset (Aljarah et al., 2020) discussed earlier has been used by Aljarah et al. (2020) to train and evaluate a hate speech detection system. However, samples from the neutral class are excluded from this study. Thus, binary classifiers are utilized with only hate and not hate samples. The following procedures are conducted during the data preprocessing phase: removing non-Arabic characters, numbers, symbols, punctuation, hashtags, URLs, and diacritics; and filtering out stopwords and negation words. In the experiments, multiple features are investigated, including BOW vectors, TF vectors, TF-IDF vectors, profile features, and emotion features. Features are selected to form several combinations during the experiments. The authors study SVM, Gaussian NB, Decision Tree, and Random Forest models in developing the system. To evaluate the performance effectively, both 10-fold cross-validation and Grid search are performed. The F1 measure is not included in the performance evaluation process; rather, the geometric mean is applied in addition to accuracy, recall, and precision. Results demonstrate the random forest model with TF-

IDF and profile features as the best one, which reports the highest geometric mean of 91.2% and the highest accuracy of 91.3%. It also shows a recall score of 94.1% and a precision score of 89.7%. Further analysis of the results demonstrates the relationship of racism, emigrant, and God word-based features to hate speech samples. The authors also highlight some challenges related to the dialectal Arabic used in social media and the shortage of Arabic resources.

Several studies have used the Twitter OSACT dataset for hate speech detection. I documented the highest macro F1 score among all hate speech experiments, as it is explained in detail in Chapter 7, which is 95.2% and achieved by applying intensive pre-possessing procedures with an SVM classifier and count vectorization of 2 to 5 characters. The experiment settings and the pre-processing procedures are identical to those discussed in Chapter 7. The result without applying any pre-processing steps shows a macro F1 score of 67%; however, it increases by 28.2% after applying all pre-processing steps. Moreover, the system achieves 95.9% in the recall, 95.2% in the precision, and 95.9% in the accuracy. These findings demonstrate the effectiveness and significance of this intensive pre-processing approach for hate speech detection, as it shows the higher performance when applied to hate speech rather than offensive language.

Djandji et al. (2020) also used the same approach they follow for offensive language detection that I discuss in the previous section, which consists of multitask learning and multilabel classification with the AraBERT model for hate speech detection. In general, results for hate speech detection show lower performance than offensive

language detection. The highest recorded macro F1 score for hate speech detection is 83.41%, achieved by multitask learning with AraBERT. The error analysis shows that confusion occurs in tweets that consist of offensive words in a non-offensive context. In addition, the authors' investigations show some mislabeled hate speech tweets, detected by their system. Generally, most of the errors are related to mockery, sarcasm, or mentioning other offensive and hateful statements within tweets.

Among the researchers that used the OSACT dataset is Hasan et al. (2020). In (Hassan et al., 2020), the basic pre-processing steps applied include removal of words with non-Arabic characters, removal of diacritics and punctuation, and reducing the repetition of the characters to only one. The authors develop various classifiers: SVM, bagged SVM, CNN-BiLSTM, M-BERT, and an ensemble classifier of SVM, bagged SVM, and CNN-BiLSTM. The best performance classifier is the ensemble classifier, which uses multiple features such as 1 to 5 n-gram characters, 2 to 6 n-gram characters, 1 to 3 n-grams words, pre-trained Mazajak word embeddings, and CNN feature extractor. The best-obtained results are 96.6% in the accuracy, 83.8% in the precision, 78.1% in the recall, and 80.6% in the macro F1 score. The authors attribute the low performance of the classifier to the fact that the dataset has low percentage of hate speech samples; only 5% are hate speech samples.

Saeed, Calders, and Kamiran (2020) use the same system architecture model that is developed for offensive language detection for hate speech detection. The authors use the OSACT dataset with multiple deep learning classifiers and traditional machine learning ensemble classifiers. The results show inferior performance for hate speech

detection in comparison to offensive language detection. The macro F1 score is 77.79% for hate speech detection, while for offensive language detection, it is 95.51%, as I mentioned in the previous section.

Similar to Saeed, Calders, and Kamiran (2020), Abu Farha and Magdy (2020) use the OSACT dataset for hate speech detection with the same system architecture previously discussed for the offensive language system that is developed by the same authors. However, the results from Abu Farha and Magdy (2020) for hate speech detection systems are lower than those achieved by offensive language detection systems. The best macro F1 score is 76%, recorded by the multitask learning model that consists of a CNN-BiLSTM classifier trained on both offensive language and hate speech tasks.

The OSACT dataset is also used for hate speech detection by Haddad et al. (2020). For the hate speech detection system, the authors apply the same pre-processing steps and classifiers that they have applied for their offensive language detection system, except the dataset is not augmented to account for the imbalance in class distribution, and the OSACT dataset is used without any modification for the number of samples for each class. The results report the Bi-GRU with attention model as the best performing model, which shows 75% for the macro F1 score, 95% for the accuracy, 75% for the precision, and 74% for the recall. The authors highlight that the hate speech detection task is a complex one; even the best performing model has difficulty distinguishing between hate speech samples and other instances of offensive language.

In Alharbi and Lee (2020), the authors also train and evaluate the hate speech model with the OSACT dataset. They use the same system they develop for offensive

language detection discussed in the previous section. However, the behavior of the two classifiers—LSTM and XGBoost—are opposites in how they behave for offensive language detection. For hate speech detection, XGBoost reveals higher performance than LSTM. The results report a macro F1 score of 74.2%, an accuracy of 96.3%, a recall of 86.4%, and a precision of 68.5% for the XGBoost model. Meanwhile, the LSTM model shows a macro F1 score of 48.7%, an accuracy of 95%, a recall of 47.5%, and a precision of 50%.

Elmadany et al. (2020) apply the same approach they follow in their offensive language experiments for hate speech detection. As we discussed earlier, Elmadany et al. (2020) augment the OSACT dataset with another manually constructed dataset with an M-BERT-based classifier as the basis for their model. For the hate speech detection system, the M-BERT that is fine-tuned on the OSACT training dataset only records the best macro F1 score, which is 70.51%. The same model also shows an accuracy score of 95.20%. It is noticeable that the behavior of the classifiers changed sharply when the tasks are switched from the offensive language detection task to the hate speech detection task.

Table 5 summarizes the studies I discuss above for hate speech detection in the Arabic language. For some studies that include various experiments for similar classifiers, I reduced them to the one that gave the best results. For example, Omar Mahmoud, and Abd El-Hafeez (2020) include three NB-based classifiers—MNB, Complement NB, and Bernoulli NB—of which I present only Complement NB in the table because it is the one that has the best results among all NB-based models.

Table 5 Qualitative analysis of hate speech literature

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Albadi et al. (2018)	Normalizing: letters, tweets attributes and numbers, lemmatization, removing stop words, non-Arabic letters, and one-letter words, handling elongation	AraVec	Twitter, 6,600 tweets	LR, SVM and GRU-RNN	71% LR, 75% SVM, 79% GRU-RNN	70% LR, 72% SVM, 77% GRU-RNN	71% LR, 72% SVM, 78% GRU-RNN	74% LR, 72% SVM, 76% GRU-RNN 8
Albadi et al. (2019a)	Adopt Albadi et al. (2018) approach	Temporal, users, and content features	Twitter, 6,600 tweets	GRU	79%	77%	84%	76%
Chowdhury et al. (2019)	Adopt Albadi et al. (2018) approach and Twitter specific technique	Word embedding, node embedding, sentence representation, and character n-gram	Twitter, 3,950 tweets	GRU, LR, SVM, LSTM, Bi-LSTM, CNN, Bi-GRU	81% Bi-GRU, CNN, and Node2Vec	89% LSTM, CNN, and Node2Vec	78% LSTM, CNN, and Node2Vec	86% LSTM, CNN, and Node2Vec
Haddad et al. (2019)	Removing RT, digits, hashtags, Emojis, usermention and non-Arabic characters	N-grams and term frequency weighting	Not available, 6,075	NB and AVM	87.9% NB, 73.9% SVM	83.6% NB, 62.2% SVM	79.8% NB, 59.9% SVM	89.5% NB, 66.5% SVM
Omar et al. (2020)	Removing non-Arabic, characters, emoji, URLs, and posts less than 2 words	Not available	Twitter, Facebook, YouTube, Instagram, 20,000 samples	Complement NB, LinearSVC, LR, Decision Tree, SGD, Ridge, Perceptron, and Nearest Centroid	97.59% NB, 97.49% SVM, 97.54% LR, 93.41% Decision Tree, 97.42% SGD, 97.07% Ridge, 96.69% Perceptron, 89.39% Nearest Centroid	97.59% NB, 97.49% SVM, 97.54% LR, 93.40% Decision Tree, 97.42% SGD, 97.07% Ridge, 96.69% Perceptron, 89.38% Nearest Centroid	97.59% NB, 97.49% SVM, 97.54% LR, 93.41% Decision Tree, 97.42% SGD, 97.07% Ridge, 96.69% Perceptron, 89.39% Nearest Centroid	97.63% NB, 97.49% SVM, 97.55% LR, 93.50% Decision Tree, 97.42% SGD, 97.07% Ridge, 96.69% Perceptron, 89.58% Nearest Centroid

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Aljarah et al. (2020)	Removing digits, hashtags, symbols, punctuation, URLs, diacritics, stopwords, negation words, and non-Arabic characters	BOW vectors, TF vectors, TF-IDF, profile features, and emotion features	Twitter, 3,696 tweets	NB, SVM, Decision Tree, and Random Forest	89.4% NB, 84.2% SVM, 87.7% Decision Tree, 91.3% Random Forest	Not available	89.4% NB, 97.8% SVM, 91.5% Decision Tree, 94.3% Random Forest	89.6% NB, 78.6% SVM, 85.8% Decision Tree, 89.7% Random Forest
Husain (2020b)	Converting emoji and emoticon to text, letter normalization, replacing repeated letter by one, dialect normalization, converting hyponym to hypernym, replacing underscore in hashtag by whitespace, removing numbers, HTML tags, more than one space, symbols, stop words, and diacritics	Count vectorizer	Twitter, 10,000 tweets	SVM	95.9%	95.2%	95.9%	95.2%
Djandji et al. (2020)	Tokenization, removing usermention, retweet, URL, diacritics, emoji and newline, replacing underscore in hashtag by whitespaces	Not available	Twitter, 10,000 tweets	AraBERT	Not available	83.41%	Not available	Not available
Hasan et al. (2020)	Removing non-Arabic characters, diacritics, punctuation, and replacing repeated characters with one	N-grams and Mazajak word embeddings	Twitter, 10,000 tweets	SVM, CNN-BiLSTM, M-BERT, and ensemble	97.1% SVM, 95.9% CNN-BiLSTM, 95.7% M-BERT, 96.6% ensemble	78.2% SVM, 70.9% CNN-BiLSTM, 73% M-BERT, 79.3% ensemble	88.8% SVM, 67.5% CNN-BiLSTM, 74.9% M-BERT, 78.7% ensemble	72.5% SVM, 76.1% CNN-BiLSTM, 72.4% M-BERT, 80% ensemble

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Saeed et al. (2020)	Normalizing letters, removing usermention, newline, emoticons, emoji, diacritics, English characters, punctuation, and URLs, and process out-of-word-embeddings-vocabulary	Not available		NB, LR, SVM, Random Forest, CNN, BiLSTM, BiGRU, CNN-BiLSTM, and ensemble	Not available	48.87% NB, 71.12% LR, 72.88% SVM, 72.88% Random Forest, 75.68% CNN, 76.63% BiLSTM, 76.63% BiGRU, 75.82% CNN-BiLSTM, 77.79% ensemble	Not available	Not available
Abu Farha Magdy (2020)	Normalizing letters, removing elongation, unknown characters, diacritics, punctuation, English characters, punctuation, and URLs	FastText and Word2Vec	Twitter, 10,000 tweets	NB, BiLSTM, M-BERT, CNN-BiLSTM, and Multitask Learning	Not available	39% NB, 67.1% BiLSTM, 71.9% M-BERT, 70.2% CNN-BiLSTM, 73.7% Multitask Learning	Not available	Not available
Haddad et al. (2020)	Normalizing letters, removing diacritics, emoticons, emoji, punctuation, non-Arabic characters, and stopwords, replacing some tokens, with Arabic words, reducing repeated letters and elongation	AraVec word embeddings	Twitter, 10,000 tweets, YouTube, not available	SVM, LR, CNN, Ridge, BiGRU,	39.1% SVM, 49.2% LR, 91% CNN, 46.2% Ridge, 95% BiGRU	31% SVM, 36% LR, 67% CNN, 36% Ridge, 75% BiGRU	46% SVM, 45% LR, 78% CNN, 57% Ridge, 74% BiGRU	49% SVM, 49% LR, 36% CNN, 51% Ridge, 75% BiGRU
Alharbi and Lee (2020)	Normalizing letters, removing punctuation, non-Arabic characters, and specific segmentation	Mazajak, AraVec, and FastText word embeddings	Twitter, 10,000 tweets	LSTM, and XGBoost	95% LSTM, 96.3% XGBoost	48.7% LSTM, 74.2% XGBoost	47.5% LSTM, 86.4% XGBoost	50% LSTM, 68.5% XGBoost

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
	for ي/O words							
Elmadany et al. (2020)	Not available	Not available	Twitter, 10,000 tweets	M-BERT	95.2%	70.51%	Not available	Not available

2.3 Cyberbullying

Haidar, Chamoun, and Serhrouchni (2017; 2018; 2019) examine cyberbullying from multiple perspectives. They developed a multilingual classification system that can process both Arabic and English texts. The authors collect a dataset from Twitter (4.93GB) based on the geographic location of posts. They extract additional posts from Facebook (0.98GB) for further evaluation after training and testing the model (Haidar, Chamoun, & Serhrouchni, 2017). They select the center of the Middle East Region by defining an area within a 10,000 kilometers' radius in each run. The collected dataset includes posts mostly from Lebanon, Syria, the Gulf Area, and Egypt. The dataset was filtered to remove all duplicate samples and samples in languages other than Arabic and English; then, they separate posts into an Arabic dataset of size 35,273 tweets and an English dataset of size 91,431 tweets. Datasets were manually labeled with either "yes" for bullying instances or "no" otherwise. They used WEKA to preprocess texts, owing to its support for the Arabic language. Features vectors include Tweet-To-SentiStrength-Feature-Vector and converting strings to word vectors. The Tweet-To-SentiStrength-Feature-Vector was provided by using the SentiStrength tool with a customized, multi-

dialect list of profane Arabic words, which gives all terms in the list a weight of –5 (mostly negative) to use as the input to the SentiStrength tool to generate the sentiment score of each tweet. NB and SVM-based classifiers were used to develop the system. Even though the system aims to provide multilingual support, and the datasets include English tweets, the researchers train and test the model only on Arabic datasets in the study. Without using any features to the model, the results of the NB show an F1 score of 90.5%, while results for the SVM, using both Tweet-To-SentiStrength-Feature-Vector and converting strings to word vectors, show an F1 score of 92.7% (Haidar, Chamoun, & Serhrouchni, 2017).

Haidar, Chamoun, and Serhrouchni (2017) extend their study to include deep learning models (Haidar, Chamoun, & Serhrouchni, 2018). They develop an FFNN, also known as multilayer perceptron, for Arabic cyberbullying detection (Haidar, Chamoun, & Serhrouchni, 2018). The FFNN has interconnected layers with the data flows in the direction from the input layer to the output layer, without having the data flow back (Haidar, Chamoun, & Serhrouchni, 2018). The same dataset previously constructed and used in their earlier study is employed again in this study, with some modifications. These changes consist of removing all hyperlinks, non-Arabic characters and emoticons, and binarized labels (1 for “Yes” and 0 for “No”). In addition, the dataset was split into a small dataset with 4,913 (1,688 bullying instances) tweets and a large dataset with 34,890 (3015 bullying instances) tweets (Haidar, Chamoun, & Serhrouchni, 2018). Tweets were tokenized and all unneeded characters were removed. Then, one hot encoding was used to create word embeddings. The best result shows an accuracy of 94.56%, which is

reported when using FFNN with 2 epochs, 7 hidden layers, and batch size of 16 (Haidar, Chamoun, & Serhrouchni, 2018). Haidar, Chamoun, and Serhrouchni (2019) further improve their system by applying some ensemble ML methods. They used a dataset of 31,891 unbullying tweets and 2,999 bullying tweets, manually labeled using two classes: 'bull' for bullying instances and 'None' for other instances. Tweets were preprocessed by removing hyperlinks, non-Arabic characters, hashtags, retweets, and stemming words after tokenization. An algorithm for generating word embedding was used to create features (Haidar, Chamoun, & Serhrouchni, 2019). Three ML models were explored: (1) stacking with simple linear regression as the meta-learning mechanism using a set of single learner classifiers that includes random forest, SVM, kNN, Bayesian logistic regression, and SGD; (2) three single learners with boosting as the meta-learning mechanism, which uses NB, SVM, and nearest neighbor as the single learner classifiers; and (3) similar to (2) but with Bagging. Results show better performance of the first model, stacking, compared to the performance of the other models, boosting and bagging (Haidar, Chamoun, & Serhrouchni, 2019). In addition, comparing the results from applying ensemble ML methods to the previous result recorded using NB single learner shows much better performance when using ensemble ML methods. The earlier single learner, NB, achieved a 90.05% F1 score, while the ensemble meta-learner, Stacking, shows a 92.6% F1 score (Haidar, Chamoun, & Serhrouchni, 2019). Table 6 reviews the previous studies we discussed above in this section.

Table 6 Qualitative analysis for cyberbullying literature

Ref.	Pre-processing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Haidar et al., (2017)	Not available	TweetToSentiStrength- FeatureVector, converting strings to word vectors	Twitter, 35,273 tweets	NB and SVM	Not available	90.5% NB, 92.7% SVM	90.9% NB, 94.1% SVM	90.1% NB, 93.4% SVM
Haidar et al., (2018)	Removing hyperlink, and non-Arabic characters, binarize labels	One hot encoding for word embedding	Twitter, 4,913 tweets (small dataset) 34,890 tweets (large dataset)	FFNN	94.56%	Not available	Not available	Not available
Haidar et al., (2019)	Removing non-Arabic characters and tweet attributes, stemming	AraVec trained on Twitter dataset and Skip-gram model	Twitter, 31,891 unbullying tweets and 2,999 bullying tweets	Ensemble ML (Stacking, Bagging, and Boosting)	Not available	92.6% (Stacking)	93.8% (Stacking)	93.3% (Stacking)

2.4 Adult Content

Researchers from the previous study on detecting abusive Arabic Twitter accounts to enhance their system with further investigations into the specific type of abusive content (Abozinadah, 2017; Abozinadah & Jones, 2017). They develop a model to detect Twitter user accounts that are posting adult content in the Arabic language. They use a statistical approach to analyze tweets, which consists of three main components: (1) PageRank (PR) algorithm to distinguish the relevant words based on edge weight; (2) Semantic Orientation (SO) algorithm to identify words co-occurrence relationships based on Point-wise Mutual Information (PMI) and Information Retrieval (IR); and (3) basic statistical measurements to define tweet contents, such as the number of hashtags, mentions, pictures, URLs based on computing minimum, maximum, standard deviation,

average, and totals (Abozinadah & Jones, 2017). They randomly select 2,500 Twitter accounts from their previously constructed dataset, then narrow down their selection to 406 non-abusive accounts and 406 abusive accounts, with 50 tweets for each account randomly selected based on the labels provided from the 2,500 accounts. Pre-processing steps include removing all non-Arabic words, all symbols, all digits, all stop words, all sequences of letters in words except the name of God (Allah), all diacritics, all extra whitespace, and correcting misspelled words (Abozinadah & Jones, 2017). They develop an SVM-based classifier with 10-fold cross-validation and feature selection. The results demonstrate better overall performance and accuracy for the SVM-based model that follows the statistical approach than the previous model that uses the BOW approach. The statistical approach with the three added features—PR, SO, and necessary statistical measurements—shows an accuracy score of 96.4% (Abozinadah & Jones, 2017).

Mubarak, Darwish, and Magdy (2017) propose a system to detect vulgar and pornographic obscene speech in Arabic social media based on a list-based approach. They used a Twitter dataset consisting of 175 million tweets to extract a list of seed words for obscene phrases through manual assessment. The seed-words list is used to construct three sub-lists of obscene words, phrases, and hashtags from the same dataset. Moreover, they use another list of obscene phrases provided by TweetMogaz, along with multiple measurements, including the Log Odds Ratio (LOR) for uni-grams and bi-grams. Both intrinsic and extrinsic evaluations were applied to evaluate the two lists. The intrinsic evaluation contains manual coding for a set of 100 words, which were randomly-selected from each list to be marked as either obscene or not. The extrinsic evaluation

consists of recall, precision, and F1 measure using a dataset that they develop for this evaluation. They select ten Egyptian Twitter users listed on SocialBakers as the top controversial Twitter users. Then, they extract 100 tweets with at least ten replies for each user. The final dataset has 100 original tweets and 1,000 replies. Each tweet, along with its replies, was submitted to CrowdFlower to be coded by three annotators from Egypt using three classes: obscene, offensive, and clean. The inter-annotator agreement score is 84%. They develop a linear model to evaluate each labeled tweet; if a match with a phrase from the list occurs, it predicts a label of obscene for that tweet. Results, among all lists, record 60% as the highest F1 score, which demonstrates the inefficiency of using a list-based approach. The list-based approach is very limited and not a right choice for an obscene detection system.

Alshehri et al. (2018) propose a system to detect adult content on Twitter as well as user accounts that spread this type of content within the Arabic Twitter-sphere. To achieve their goal, they collect a set of 100 hashtags, which were used as parameters to Twitter REST and streaming APIs to get a set of tweets that forms their dataset. They apply some filtering steps to remove duplicates from the dataset until they reach a total of 200,000 Arabic tweets. From this dataset, they search for the users who post at least two tweets within the same dataset and create another dataset of 20,621 Twitter users. Hence, they use the second dataset to extract the timeline from 11,648 users, including up to 3,200 tweets, thus creating a third dataset, the timeline dataset that consists of 8.6 million tweets. They analyze the network structure of account users that spread adult content by applying some statistical measurements (e.g., number/median/mean/mode of followers,

number/median/mean/mode of following). The result of the analysis records that the average adult content-spreading user posts around 914 tweets, use 1.45 hashtags in each tweet, has around 7,489 followings, and has 850 followers in their network. In addition to analyzing the network structure, they do in-depth analysis for usernames, profiles, tweets media (e.g., URLs), and geographical distribution. They develop an SVM-based classifier that trained using a subset of the dataset containing two classes; "positive" consists of 2,500 adult content users, each having at least 500 tweets and another "negative" one with the same size as regular users. The implemented pre-processing techniques include removing all hashtags. They developed two features, BOW and Bag-Of-Means (BOM), and use them to explore their model with multiple sizes of tweets per user. Results highlight the best performance when using the BOM feature and 250 tweets per user, which gives an accuracy score of 79%. Table 7 summarizes the studies covering adult content detection we discussed above.

Table 7 Qualitative analysis for adult content literature

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Abozinadah and Jones (2017)	Removing non-Arabic characters, stop words, sequences of letters in the words except the name of God (Allah), correcting misspelled words	PR, SO, basic statistical measures, and feature selection	Twitter, 812 tweeter accounts	SVM	96.4%	96.4%	96.4%	96.4%
Mubarak et al. (2017)	Normalizing letters and decorative characters, proper	Not available	Twitter, 1,100 tweets	List-based Model	Not available	60%	45%	98%

Ref.	Preprocessing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
	segmentation of hashtags and URLs							
Alshehri et al. (2018)	Removing hashtags	BOW and BOM	Twitter, 5,000 users with at least 500 tweets per user	SVM	79% BOM, 250 tweet per user	78% BOM, 250 tweet per use	Not available	Not available

2.5 Violence

Abdelfatah, Terejanu, and Alhelbawy (2017) propose an unsupervised model to detect violent content on Twitter automatically. A dataset collected from Twitter was used that consists of 16,234 tweets coded by at least five annotators. The labeling scheme has two classes: violence or not violence. Tweets were pre-processed by removing Arabic stop words and URLs. The MADIMARA tool kit was used during pre-processing to extract glosses and tokens. After that, features were generated using TF-IDF weights. Textual data are very high dimensional; thus, it is good practice to use a dimensionality reduction technique before applying an unsupervised classifier since it does not depend on labeled data to learn features that support the classification decision. In this study, two dimensionality reduction techniques were applied, the Gaussian Process Latent Variable Model (GPLVM) and the Principle Component Analysis (PCA), to see which one gives better results. The output from the dimensionality reduction step is used to feed a k-means clustering model initialized with two clusters. Multiple combinations of pre-processing steps and dimensionality reduction techniques were explored. Results

demonstrate that when PCA is used with glosses, k-means reports an F1 score of 55%, while when the PCA is used with tokens, it reports 71%. The GPLVM shows an F1 score of 58% when used with glosses and 56% when used with tokens. It also shows that when using the PCA for tokens, it becomes very sensitive to data representation and noise in data.

Kaati et al. (2015) study terrorism-related content in Twitter, precisely, detecting Twitter accounts involved in media Mujahideen (also called supporters of jihadist groups) who use Twitter to spread propaganda content online. Kaati et al. (2015) develop a multilingual system that trains on English and Arabic datasets. The datasets used in this analysis include: (1) TWEEP-PRO-E (93 accounts) for English users who support terrorism; (2) TWEEP-PRO-A (81 accounts) for Arabic users who support terrorism; (3) TWEEP-RAND-E (742 accounts) for English users who do not support terrorism; (4) TWEEP-RAND-A (256 accounts) for Arabic users who do not support terrorism; (5) TW-PRO-E (27,753 tweets) for English tweets with content supporting terrorism; (6) TW-PRO-A (16,000 tweets) for Arabic tweets with content supporting terrorism; (7) TW-RAND-E (60,000 tweets) for English tweets without content supporting terrorism; and (8) TW-RAND-A (45,013 tweets) for Arabic tweets without content supporting terrorism. The seed words used in collecting these datasets are taken from a forum called “Shumukh al-Islam", which is known to be used by individuals who support terrorist groups. Then, they manually collect some Twitter users and hashtags, which were all used in extracting the datasets. Each dataset is split into training, validation, and testing datasets. Two types of features were applied. The first type is data independent features,

including stylistic features, time features, and emotional words. The second type is data dependent features, including most common hashtags, most common word bigrams, most common letter bigrams, and most frequent words. They use the Ada R package to create an AdaBoost classifier, which is a machine learning model that uses a meta-learning boosting algorithm to produce accurate predictions. They do multiple experiments using different combinations of features and parameters. When using the classifier for predicting the English user's category, terrorism supporter or not, results show similar performance in all combinations of data dependent or independent features with scores around 100% in all measurements. The Arabic users report the highest accuracy score of 98.48% with data dependent features only. These results reveal that it is hard to interpret the findings of this study regarding user accounts accurately because the datasets are very small. On the other hand, Tweet-based datasets are large enough to train the same model. Results for English and Arabic tweets report the highest accuracy score of 99.51% for the English dataset and 86.38% for the Arabic dataset when both features, data independent and data dependent, are used. This study depends on the accuracy score to evaluate the model's performance, which might not be the best metric to use as the datasets are imbalanced, and the accuracy score does not consider this criterion within the calculation.

Magdy, Darwish, and Weber (2017) build a classifier to identify terrorism-related content written in Arabic that supports the Islamic State of Iraq and Syria (ISIS) on Twitter. The focus of this study is on understanding the violent behavior, in terms of expressing support to ISIS, of the ideological supporters of ISIS rather than detecting the

actual fighters. They use a simple linear SVM-based model with BOW in developing the classification model. Their model gives a significant F1 score of 87%, which demonstrates an acceptable prediction for future support or opposition of ISIS based on pre-ISIS period data. Firstly, they extract a dataset from the Twitter Streaming API based on two parameters: tweets written in the Arabic language and tweets including terms from a set of keywords that represent ISIS (e.g., داعش /da'esh/an Arabic acronym for ISIS, الدوله الإسلاميه/Aldawla Alislamiya/Islamic State). Secondly, 1,000 tweets were selected from the initial dataset to be coded by one native Arabic annotator as positive (pro-ISIS), negative (anti-ISIS), or neutral (news or spam). Thirdly, the labeled dataset was used to identify a set of Twitter users, which were used as input to the Twitter REST API for retrieving their profile information along with their latest tweets. The data analysis approach of the study depends on temporal information that lasts for a period of about two months to ensure having enough information for pre-ISIS and post-ISIS periods. During this time, multiple ISIS supporter accounts were terminated by Twitter; the study started with 11,332 pro-ISIS users and 45,628 anti-ISIS users, but due to Twitter termination, they narrow down the anti-ISIS accounts to have a balanced dataset, ending with 7,225 accounts from each class, which were used to train and validate the model. The preprocessing phase includes text normalization and handling word elongations. Moreover, for each Twitter account, they combine all its latest tweets into a single document, which is later used to create features. Features include BOW for unigrams, hashtags, and user mentions. The model uses an SVM-based classifier with 10-fold cross-

validation and reports an average of 87% for all performance evaluation scores of the recall, precision, and F1.

In another multilingual study, some deep learning methods were applied to identify if the online text contains extremist or terrorism-related content (Johnston & Weiss, 2017). The Johnston and Weiss (2017) study is driven by the assumptions that extremist and terrorist groups are using social media to spread propaganda, recruit new members, instruct users on how to conduct terrorist attacks, and communicate with their members. A team of 40 people, including experts in the Arabic language, Islam, and Middle Eastern culture, collect and label the dataset using two labels; "benign" and "extremist". They use multiple multilingual websites that belong to terrorist groups to collect their extremist data and collect the non-extremist data from different domains of news articles (e.g., tennis, Syrian civil war) and the English version of the Qur'an. Having a dataset of several languages (64.97% Arabic, 19.23% English, etc.) makes it very difficult to pre-process and normalize the text as each language has different characteristics and structures. Consequently, they follow a language-agnostic approach by converting all texts to ASCII characters before pre-processing. Another difficulty in pre-processing this dataset is its variability in document length, which was tackled by splitting documents based on newline characters. The final dataset consists of 264,117 lines with 69.9% extremist instances and 30.1% benign instances. Labels were binarized, and lines were converted to vectors using Doc2Vec, which were used to train a deep CNN model consisting of 2 hidden layers, each with 42 and 14 neurons. The network uses Adam optimizer, categorical cross-entropy loss function, and Parametric Rectified

Linear Unit (PReLU) for all nodes except the output layer, which uses a Softmax function for the predictions. Their system reports significant results; the recall score is 95.6%, the precision score is 95.9%, the F1 score is 93%, and the accuracy score is 93.2%. Table 8 reviews the previous studies for violence detection we discussed earlier in this section.

Table 8 Qualitative analysis for violence literature

Ref.	Pre-processing	Features	Dataset	Classifier	Accuracy	F1	Recall	Precision
Kaati et al. (2015)	-	Data independent features, data dependent features	Twitter, 93 English accounts pro-terrorist,81 Arabic accounts pro-terrorist, 742English accounts anti-terrorist, 256 Arabic accounts anti-terrorist,27,753 English tweetspro-terrorist, 16,000Arabic tweets pro-terrorist, 60,000English tweets anti-terrorist, and 45,013Arabic tweets anti-terrorist	AdaBoost	100% English accounts, 98.48% Arabic accounts, 99.51% English tweets, 86.38% Arabic tweets	Not available	100% English &Arabic accounts, 99.92% English tweets, 86.35% Arabic tweets	100% English account, 90% Arabic accounts, 98.53% English tweets, 56.91% Arabic tweets
Magdy et al. (2016)	Text normalization, handling word elongations, for each Twitter account, all its latest tweets were combined into a single document	BOW for individual terms, hashtags and user mentions	Twitter, 7,225 anti-ISIS Twitter accounts and 7,225 pro-ISIS Twitter accounts	SVM	Not available	87%	87%	87.2%
Johnston and Weiss (2017)	Language-agnostic preprocessing approach	Dov2Vec vectors	Twitter, 264,117 Lines	Deep CNN	93.2%	93%	95.6%	95.9%

Quantitative Analysis of Studies

There is very limited research in Arabic offensive language detection, specifically in cyberbullying, violence, and adult content. I notice that there are some repetitions in the authors' names, indicating the limited number of researchers in this domain. The most frequent author is Hamdy Mubarak, with several studies and datasets covering general offensive language, hate speech, and adult content. I quantitatively discuss and analyze my findings in the following paragraphs.

1. Trends in Literature

There has been no research on Arabic offensive language detection before 2015. Figure 16 shows trends in research covering different types of Arabic offensive content. Most of the studies that are published in 2015 to 2017 cover violence-related topics, such as terrorism and extremism. This finding might be related to the fact that wars and conflicts in the Middle East region were very dominant during that time. Adult content starts gaining popularity in 2017, and a broader range of offensive forms are studied in 2018. Cyberbullying has a stable number of studies over the years 2017 to 2019 and all are provided by the same authors. Hate speech becomes more popular in 2019, a year after publicly releasing the first Arabic religious hate speech corpus in 2018. Moreover, after the release of the OSACT dataset in 2020, the number of offensive language studies increases by nine, and the number of hate speech studies increases by five from the previous year.

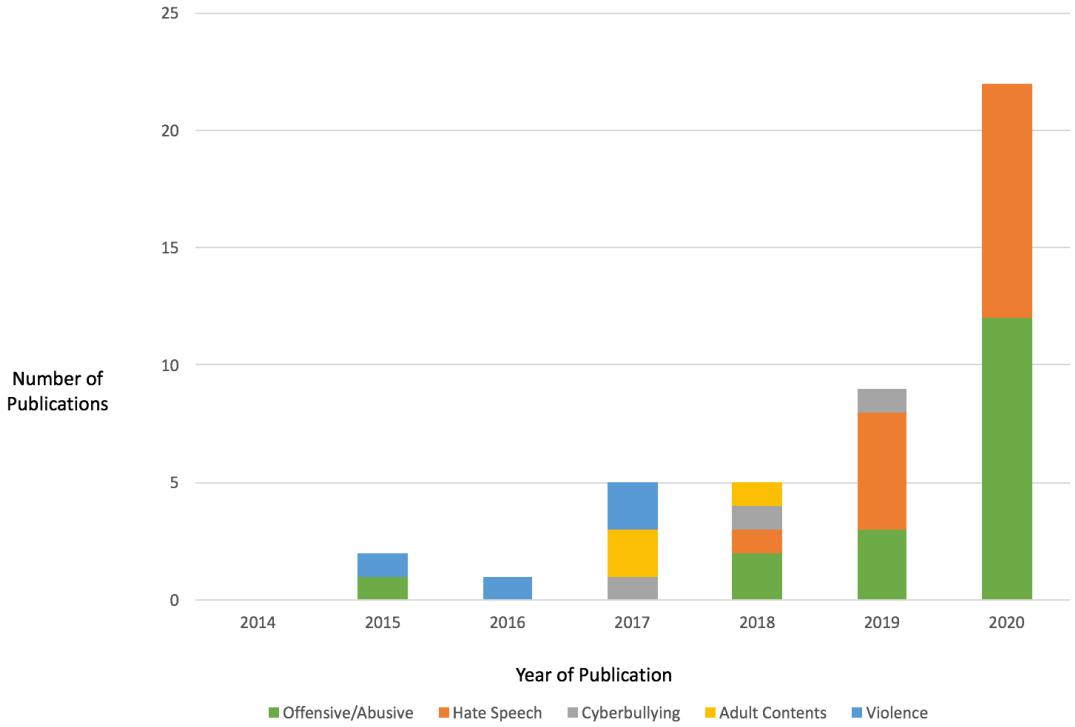


Figure 16 Trends in Arabic offensive language detection literature

Figure 17 shows an overall distribution for the categories of offensive language I found. There are few studies on cyberbullying with limited datasets; I found only three. All of them are authored by the same researchers, and all of them use the same dataset, which was not publicly available (Haidar, Chamoun, & Serhrouchni, 2017; 2018; 2019). Adult content studies are very limited in terms of the number of studies and the year of publications. Three adult content studies are found that are made by various researchers, each using its own dataset. The first study was published in 2017, and the last in 2018. The total published violence studies are slightly more prevalent than cyberbullying and adult content studies; I find four studies during our search. They have mostly even

distribution among most years, from 2015 to 2017. On the other hand, hate speech and offensive language are the most dominant categories for 2019 and 2020.

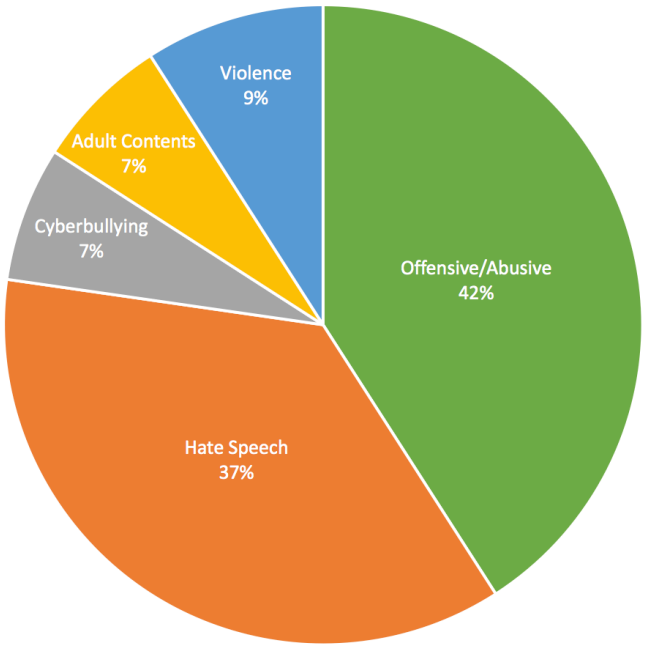


Figure 17 Categories of offensive language

2. Datasets Characteristics

Figure 18 shows the distribution of the dataset sources among the studies. Twitter is the predominant platform of the surveyed studies, followed by YouTube, Facebook, websites, and Instagram. Only one study uses terrorism-related content manually-collected from multiple websites and online outlets (Johnston and Weiss, 2017). Twitter is a popular social media platform among most countries, and it has a very flexible API that can be used to extract datasets based on multiple criteria, such as keyword, tweet ID,

and user ID. In addition, Twitter has fewer privacy constraints in comparison to the other commonly used social media platforms. These features could be the reason why most of the studies are using datasets from Twitter. YouTube has an API that allows users to extract a list of comments based on a channel ID or a comment ID of the parent comment. The dataset sizes vary; some studies use a very small dataset, which gives questionable results as their model cannot generalize. For example, Kaati et al. (2015) use a dataset of Twitter accounts consisting of 81 and 93 samples from each class and report 100% in accuracy, recall, and precision, which make for some model generalizability issues. Figure 18 also shows the ranges of dataset sizes, including both posts-based datasets and user accounts-based datasets. Most of the studies depend on datasets that range in size from 10,000 to 15,000 samples.

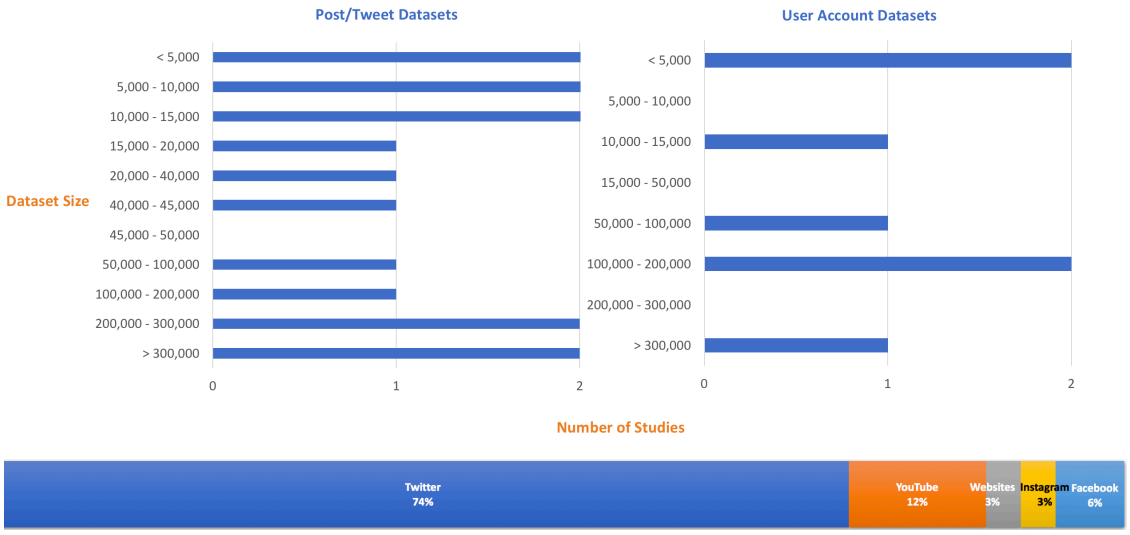


Figure 18 Sources of datasets used among literature

3. Classification Model

The majority of the studies follow the classical pipeline of supervised machine learning, starting from constructing the dataset, pre-processing, feature extraction, building the model, and evaluating the performance of the model. Only one study uses an unsupervised classification method and reports relatively lower performance with an F1 score of 60% (Mubarak, Darwish, & Magdy, 2017). The rest of the studies use either a single-learner machine learning classifier (e.g., SVM, LR, NB) or a deep learning classifier (e.g., CNN, GRU, FFNN). The SVM-based classifier is the dominant model among all literature. The next model in use after the SVM are the CNN, the NB, and the GRU-based classifiers. Several studies apply ensemble methods; 10% of the experiments performed by these studies include ensemble classifiers such as AdaBoost, random forest, and bagging. Figure 19 illustrates the classification models used in the literature. It is vital to notice that I consider all experiments conducted by each study based on the classifiers. Thus, some studies study several classifiers; I consider all of them in my analysis.

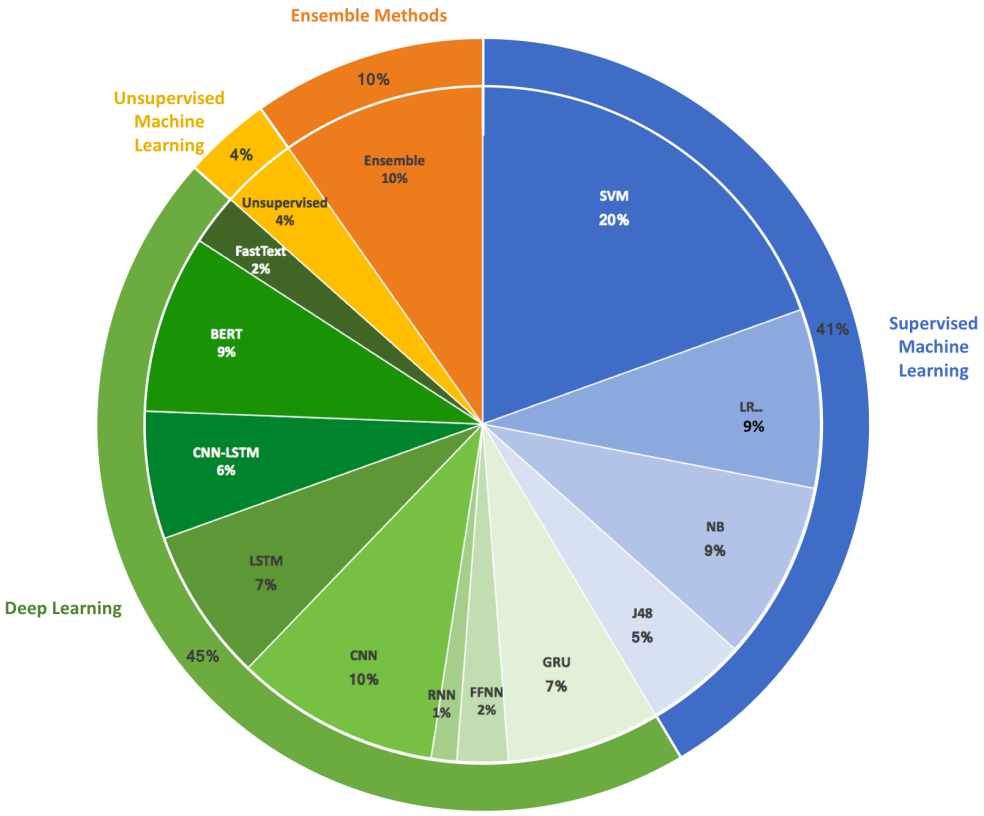


Figure 19 Classification models used among literature

4. Performance Evaluation

Multiple evaluation metrics are used in the literature, including accuracy, F1, precision, recall, and confusion table. In my quantitative performance analysis, I exclude results obtained by the Kaati et al. (2015) study due to the very small dataset used in developing the model. The highest recorded accuracy is 98.7%, which is achieved by Omar, Mahmoud, and Abd El-Hafeez (2020) using an RNN-based classifier to detect hate speech on a multi-platform dataset. The same study also has the second-highest accuracy score, 98.35%, reported by the CNN-based model. Simple evaluation-based

only on accuracy might not be reliable, as most of the studies are using the imbalanced datasets in building their models and the accuracy measurement is very sensitive to imbalanced distribution of classes. Thus, investigating the F1 score might be better for this domain since online offensive language often occurs less frequently than typical content. The F1 score considers the tradeoff between recall and precision and, consequently, it gives a more accurate evaluation of the classification performance. The RNN-based model and the CNN-based model from the Omar, Mahmoud, and Abd El-Hafeez (2020) study are also the top two systems according to the F1 score, with 98.7% and 98.35%, respectively. Table 9 highlights some statistical details for performance measurements for each form of offensive language, separately, as well as for all of them after removing outliers; 100% scores from all metrics. I include all experiments performed by all studies in my analysis. The number of experiments per study ranges from one experiment (Alakrot, Murray, & Nikolov, 2018a; Haidar, Chamoun, & Serhrouchni, 2018, 2019; Johnston & Weiss, 2017; Magdy, Darwish, & Weber, 2017) to 26 experiments (Chowdhury et al., 2019). To deliver a better understanding of the topic, I include an in-depth analysis in Table 9 based on statistical calculations for each category of offensive content.

Table 9 Performance evaluation from literature

Offensive Form	Measurement	Maximum	Average	Minimum	Standard Deviation	Number of Studies
Offensive/ Abusive	Accuracy	94.3%	89.7%	77.7%	3.5%	41
	F1	95.51%	76.4%	21%	15.94%	49
	Recall	91.5%	82.97%	64%	5.87%	40
	Precision	93.5%	84.01%	63%	6.61%	40
Hate Speech	Accuracy	98.7%	81.14%	69%	17.28%	77
	F1	98.7%	73.36%	31%	15.24%	96
	Recall	98.7%	79.9%	45%	14.02%	73
	Precision	98.7%	73.52%	24.92%	15.36%	73
Cyberbullying	Accuracy	94.56%	94.56%	94.56%	0	1
	F1	92.7%	91.94%	90.5%	1.24%	3
	Recall	94.1%	92.94%	90.9%	1.77%	3
	Precision	93.4%	92%	90%	1.72%	3
Adult Content	Accuracy	96%	71%	53%	16%	12
	F1	96%	63%	38%	19%	17
	Recall	96%	58%	41%	25%	7
	Precision	98%	94%	89%	4%	7
Violence	Accuracy	98.48%	90.64%	82.4%	6.23%	7
	F1	93%	64.6%	54%	14.33%	10
	Recall	100%	79.83%	55%	15.3%	16
	Precision	95.9%	63.22%	46%	18.96%	16
Overall	Accuracy	98.7%	77.27%	53%	27.39%	138
	F1	98.7%	69.06%	38%	24.76%	225
	Recall	98.7%	76.72%	41%	21.31%	137
	Precision	98.7%	72.1%	89%	22.16%	139

Discussion

Datasets and Lexicons Implications

Alakrot, Murray, and Nikolov (2018a) pinpoint the importance of considering four principles when constructing a dataset. Firstly, availability; a dataset taken from a publicly available platform, thus reflecting their behavior in normal condition. Secondly, representativeness; this principle is related to availability. When the platform is public, it covers a larger segment of people and is, consequently, more representative of real-life situations. Thirdly, heterogeneity; the dataset covers diverse text in several writing styles reflecting the heterogeneity of content on the web. Fourthly, balance; the dataset should have equal instances of all label classes. Among all studies, few researchers consider these principles. One widespread drawback for the majority of datasets is imbalanced classes. Haidar, Chamoun, and Serhrouchni (2017) discuss this limitation of their imbalanced cyberbullying dataset. They highlight a problem that arises from the attempts by multiple other researchers to mitigate this issue, which is multiplying the samples of the rare class until it becomes equal to the size of the other class. This way of mitigating the imbalanced dataset limitation is very questionable because it creates a training dataset that is far from the real world (Haidar, Chamoun, & Serhrouchni, 2017). In (Haddad et al., 2020), the authors propose another method to deal with imbalanced classes by augmenting the dataset with another dataset to add more samples from the minority class. This approach could be misleading, as ensuring the equivalence of the main and augmented datasets—in terms of the source platform, annotation process, and type of the language used—is very complicated.

I notice that the OSACT (Mubarak et al., 2020) dataset becomes a benchmark dataset among the general offensive language and hate speech studies. Having a benchmark dataset supports better evaluation of the studies, as it allows for comparative analysis among the studies. Earlier researchers are using diverse datasets; each study builds its own dataset to serve its targets. However, after the release of the OSACT dataset in 2020, most studies adopt it to develop their system. More than ten studies are using it. Table 10 shows an overall picture of datasets used among surveyed studies; it includes information about the datasets that are either being used in building models or have been constructed by researchers to add more resources to research in this domain. From the table, it is noticeable that only two datasets have hierarchical labeling schema (Albadi, Kurdi, & Mishra, 2018; Mubarak et al., 2020). Having multiple hierarchies might add some complexity. However, it reduces ambiguity for annotators if it is within a limited number of levels (such as two). Inter-annotator agreement score is important to ensure the quality of the annotating system, but not all researchers considered such a measurement. This could be because some researchers assigned the labeling task to only one annotator, which might raise some bias concerns as well. These findings illustrate the need to develop a benchmark dataset that satisfies all four principles: availability, representativeness, heterogeneity, and balance, with high annotation quality and a reasonable size for cyberbullying, adult content, and violence detection.

Table 10 Arabic datasets for offensive language detection

Ref.	Dataset Purpose	Source	Size	1st Level Labels	2nd Level Labels	Inter-annotator Agreement	Public/ Private
Albadi et al. (2018)	Religious hate speech dataset	Twitter	6,600 tweets	Hate (2,526), not hate	Muslims 9.25%, Jews 33.06%, Christians 25.02%, Atheists 23.75%, Sunnis 7.36%, Shia 31.99%, other 16.98%	81% 1st level label, 55% 2nd level label	Public
Alakrot et al. (2018a)	General offensive language	YouTube	15,050 comments	Positive (5,817), negative	Not available	71%	Private
Haidar et al. (2017)	Multilingual cyberbullying detection	Twitter	35,273 (Arabic), 91,431 (English)	Yes (2,196) for cyberbullying instances, No	Not available	Not available	Private
Mubarak et al. (2017)	Vulgar and pornographic obscene speech	Twitter	1,100	Obscene (19.1%), offensive (40.3%), clean (40.6%)	Not available	84%	Public
Mubarak et al. (2017)	Personal attack, racist, sexist, offensive, inciting violence, non-relevant, or advertising	Aljazeera.net	32,000 comments	Obscene (2%), offensive (79%), clean (19%)	Not available	87%	Public
Abozinadah et al. (2015)	Abusive Twitter accounts	Twitter	1,300,000 tweets and 350,000 Twitter accounts	Abusive and non-abusive	Not available	Not available	Private
Johnston and Weiss (2017)	Extremist or terrorism multilingual content	Websites and other multilingual online sources	264,117 lines	Extremist (69.9%), benign (30.1%)	Not available	Not available	Private
Madgy et al. (2016)	Terrorism supporter behavior	Twitter	14,450 Twitter accounts	Anti-ISIS (7,225), pro-ISIS (7,225)	Not available	Not available	Private
Kaati et al. (2015)	Multilingual terrorism supporter behavior (EN) and (AR)	Twitter	835 EN accounts,337 AR accounts, 87,753 EN tweets, 61,013 AR tweets	93 EN accounts pro-terrorist, 81 AR accounts pro-terrorist,742 EN accounts anti-terrorist, 256 AR accounts anti-terrorist, 27,753 EN	Not available	Not available	Private

Ref.	Dataset Purpose	Source	Size	1st Level Labels	2nd Level Labels	Inter-annotator Agreement	Public/ Private
				tweets pro-terrorist, 16,000 AR tweets pro-terrorist, 60,000 EN tweets, anti-terrorist, and 45,013AR tweets anti-terrorist			
Alsheh ri et al. (2018)	Adult content	Twitter	5,000 users with at least 500 tweets per user	Positive (2,500 adult content users), negative (2,500 regularusers). Each has at least500 tweets	Not available	Not available	Private
Mulki et al. (2019)	Levantine hate speech and abusive contents	Twitter	5,846 tweets	Normal (3,650), abusive (1,728), hate (468)	Not available	76.5%	Public
Haddad et al. (2019)	Tunisian hate speech and abusive contents	Not available	6,075 comments	Normal (3,834), abusive (1,127), hate (1,078)	Not available	81.82%	Public
Mubara k et al. (2020)	Hate speech and offensive language	Twitter	10,000 tweets	Offensive (1,900), not offensive (8,100), hate (50), not hate (9,950)	Not available	Not available	Public
Aljarah et al. (2020)	Hate speech	Twitter	3,696 tweets	Neutral (2,061), not hate (790), hate (843)	Not available	Not available	Private
Omar et al. (2020)	Multi-platform hate speech	Facebook, Twitter, YouTube, and Instagram	20,000 samples	Not hate (50%), hate (50%)	Not available	Not available	Private
Chowd hury et al. (2020)	Offensive language	Twitter, Facebook and YouTube	4,000 samples	Offensive (16.88%), not offensive (84.13%)	Not available	Not available	Public

As I mentioned earlier, Habash (2010) divides the Arabic dialects into seven main categories. However, Egyptian, Levantine, and Tunisian are the only Arabic dialects that

are considered in creating dialectal datasets for offensive language detection. This limited coverage of the Arabic dialects needs to be addressed by future researchers.

I also notice a common limitation among lexicon lists provided by the surveyed studies; most of them only consider single words in developing their list. A better approach might be to consider concepts and n-grams, similar to the approach in Mustafa et al. (2017), a study for Urdu offensive language detection on Twitter. Mustafa et al. (2017) use some data mining techniques to find hidden patterns and relationships of terms to their actual class, including Top-K Sequential pattern mining (TKS), Co-occurrence MAP (CM-SPAM), and Equivalence class-based sequential Rule Miner (ERMiner); using variations of n-gram from one word to three words.

System Pipeline Implications

I aggregate findings of all studies regarding pre-processing, feature extraction, and classification models into one table to show an overall view of what researchers have studied in this domain thus far (see Table 11). As can be noticed, the pre-processing steps for Arabic text are not like those for English (e.g., lowercase letters, lemmatization). Some techniques that have been used are language-specific, such as removing Kashidas and diacritics, while other techniques are language agnostic, such as converting text to ASCII characters and splitting text based on newline characters. However, models and features applied in the literature are similar to those developed by other studies covering languages other than Arabic, except for the AraBERT model, AraVec word embeddings features, and Mazajak word embeddings. This finding indicates the importance of

studying new approaches for feature extraction and model development to work better for the Arabic language.

Table 11 Explored pre-processing steps, features, and classification models

Attribute.	Previous Arabic Offensive Language Studies
Pre-processing	Stemming, lemmatization, tokenization
	Replacing Persian and Urdu letters with equivalent Arabic letters
	Correcting commonly mistaken letters that have phonetic similarity (e.g., dhaadh, dza'a)
	Normalizing: Alif, Alif Maqsura, Ta Marbouta, links, user mentions, and numbers
	Removing: stopwords, numbers, diacritics, punctuation, emojis, non-Arabic characters, emoticons, one-letter words, Kashidas (tatweel), sequences of letters in words except the name of God (Allah), extra whitespace, hashtags, retweets, HTML tags, and URLs
	Normalizing hashtags by deleting underscores and the # symbol
	Emojis and emoticon conversion to textual label
	Dialect normalization for selected nouns
	Hyponym conversion to hypernym
	Handling elongated words
	Correct misspelling
	Language-agnostic approach (e.g., convert to ASCII characters, split based on newline characters)
	For each Twitter account, all its latest tweets were combined into a single document
	Converting tweets to social network graphs
	Splitting OOWEV words into 2 tokens whenever the first character is Wa, Fa, or Sa
	Segmenting word starts with ﻳ /yA/O into 2 words
	Converting special tokens (e.g., URL) to their Arabic equivalent
Features	TF-IDF weighting, term weighting
	Doc2Vec, Node2Vec, Word2Vec
	BOM, BOW
	One hot encoding word embedding
	AraVec and Mazajak word embeddings
	BoW for individual terms, hashtags, and user mentions
	Profile-based features, tweet-based features, social graph features
	PR, SO
	Sentence representation
	Skip-gram model
	Character n-gram
	Sentiment analysis

Attribute.	Previous Arabic Offensive Language Studies
	Data independent features (stylistic features, time features, and emotion words)
	Data dependent features (most common hashtags, most common word bigrams, most common letter bigrams, most frequent words)
Classification Models	SVM, LR, NB, Decision Tree, SGD, Ridge, Perceptron, Nearest Centroid
	RNN, GRU, CNN, FFNN, LSTM, Bi-LSTM, Bi-GRU, various ANN with attention mechanism, CNN-LSTM, CNN-BiLSTM
	FastText, M-BERT, AraBERT
	Bagging, Random Forest, XGBoost, Adaboost
	List-based model, GPLVM or PCA followed by k-mean clustering

Gaps in Literature

The available studies regarding offensive language detection for the Arabic language have several limitations, some of which I mention above. Furthermore, some forms of the Arabic language are not studied in any of these literature in detecting offensive language (such as CAL, Arabizi, and mixed code Arabic language). All studies cover dialectal Arabic, which is mostly used in social media. Only three studies consider the variation in dialects and its effect on identifying offensive content: Mubarak et al. (2017), Mulki et al. (2019), and Haddad et al. (2019), which narrow down their scope to particular dialect only. Haidar, Chamoun, and Serhrouchni (2017) define specific dimensions for the area of the Middle East as a parameter in selecting their dataset in order to extract tweets from various regions with diverse dialects, including Lebanon, Syria, the Gulf Area and Egypt. However, they do not consider the effects of this considerable variation in dialects when they annotated their data. Kwok and Wang (2013) and Sap et al. (2019) investigate the effect of social context on offensive language

detection studies and report racial and cultural biases as the most common challenges in studying the offensive language that occurred during dataset labeling. For example, the word "عافية / Afiah" means health in Gulf, Egyptian, Iraqi, and Levantine dialects, while it means fire in Moroccan dialect. Another example, the word "لبوة / Labowa" means lioness and is used accordingly in most Arabic dialects, except in Egypt where it is used to describe immoral lady. Thus, it is crucial to have annotators representing the same dialect and culture of the content providers.

In addition, most datasets come from one platform. Only two multiplatform datasets are found among the surveyed literature (Omar, Mahmoud, & Abd El-Hafeez, 2020; Chowdhury et al., 2020). Previous studies call attention to the specificity of offensive language to each platform. Thus, a model developed using a Twitter dataset cannot be generalized to Facebook, Instagram, etc. In addition, this specificity might be related to the age of users, as some platforms are popular among specific age groups. Therefore, the type and language of offensive language might reflect their age. For instance, LinkedIn is not popular among teenagers, and thus offensive language on that platform cannot be considered representative of offensive language used by teenagers. Some platforms are not very popular in particular countries while popular in others. Thus, offensive language in one platform is not inclusive to the dialect and culture of the people of the missing countries. For example, Reddit is a good representative of Americans but not of people from the Arabian Gulf. Moreover, some platforms are commonly used by a specific gender, making the model biased toward that gender. Consequently, it might be

better to construct a dataset with texts from multiple sources to ensure the generalization and inclusion of the proposed solution.

Emojis and emoticons are commonly used by users on social media to express their feelings and attitudes toward others, topics, objects, etc. Treating them with equal importance to other textual content can support the detecting of offensive language. However, I find few studies that consider emojis or emoticons within the literature; they mostly remove them during pre-processing step.

The main goal is to detect offensive language, but that does not mean neglecting the not offensive language. All studies focus only on analyzing the offensive samples deeply, without further analysis of the not offensive samples. Trying to analyze the not offensive samples further by having a second level of labeling hierarchy might increase the efficiency of the system. Given that both offensive and not offensive samples were gathered using the same initial extraction criteria, which indicates that they share some similarities, there is a need to identify other similar classes that are not offensive.

Chapter Summary

In this chapter, I review studies covering Arabic offensive language detection, including multiple categories of offensive language such as general offensive, hate speech, cyberbullying, adult content, and violence. The main objective of this review is to investigate the SOTA research in Arabic offensive language detection that applied NLP approach in their studies. My surveying method includes multiple steps starting with keywords selection, following by search for studies, recursive search, filter studies, qualitative and quantitative analysis, and concluding with identifying gaps in the

literature. Findings from this survey indicate the importance of developing innovative approaches for feature extraction and model development that could work better for the Arabic language. This chapter includes a short section that reviews the available Arabic offensive language datasets; however, the next chapter expands it and provides insights into the content and context of the datasets.

EXPLORATORY DATASET ANALYSIS

The Holy Prophet Mohammad (peace be upon him and his holy progeny) said:
"إنّ اللّه حَرَّمَ الجَنَّةَ على كُلِّ فَحّاشٍ بَذِيءٍ، قَليلِ الحَياءِ، لا يُبالِي ما قالَ ولا ما قيلَ لَهُ."
"Allah (God) has prohibited Heaven to those who use obscenity, vulgarity, lack shame, and are not concerned about what is said or not said to them."

- Al-Kafi, volume 2, page 323, number 3; Mizan ul Hikmah, page 645

In the previous chapter, SOTA research in the Arabic offensive language is presented with emphasis on the approaches used in developing classification models. This chapter complements the previous one by adding more insights towards resources and datasets used in Arabic offensive language research. Specifically, I survey several available open-source Arabic offensive language datasets to provide a comprehensive overview by conducting in-depth Exploratory Data Analysis (EDA). The EDA includes a statistical analysis, a textual analysis, and a contextual analysis for all datasets to investigate the content from multiple dimensions. Some visualization tools are used to better understand the content and context of the data used. The study ends with a summary of the results.

The scope of this chapter covers the following research questions:

- What is the content of the available Arabic offensive language datasets?
- What are the limitations of the available Arabic offensive language datasets?

- How can we complement the available Arabic offensive language datasets to contribute to text classification systems?

The chapter is organized into four main sections. The methodology is described in detail in the first section. The second section presents the results, and the third section builds on the second one by discussing and synthesizing the results. In the last section, conclusions and design considerations are presented.

Methodology

I follow four main phases during the survey process. They are starting from selecting datasets, formatting datasets, analyzing datasets, and ending by summarizing and synthesizing the results. The following paragraphs describe each phase of the methodology in detail.

1. Selecting Datasets

I define a set of criteria to select the datasets: searching, formatting, and accessibility. These criteria ensure the quality of the study.

a. Defining Searching Criteria

I include datasets related to offensive language, such as hate speech, vulgar, or abusive. Only Arabic language datasets are considered, including dialectal Arabic.

b. Defining Formatting Criteria

Datasets from multiple formats were included. Most datasets are in Comma-Separated Values (CSV) file format; few of them are in Excel, Tab-Separated Values (TSV), and JavaScript Object Notation (JSON).

c. Defining Accessibility Criteria

Datasets that have been released freely online with open-source options are considered only.

2. Formatting Datasets:

The selected datasets are in heterogeneous formats and some of them include multiple descriptive attributes, such as publishing date, user profile, or number of annotators. Thus, I process them to be in minimal and consistent format.

a. Filtering Attributes

I remove all unnecessary attributes that do not serve the goal of the study. Only textual messages and labels were included. The content of textual messages was intentionally kept because all content is considered for analysis purposes; however, some datasets were provided in preprocessed format only.

b. Creating CSV Files

For each dataset, I create a CSV file to save the textual messages and labels only. This file is used for cross labels analysis and overall dataset analysis.

c. Creating Textual Files

For each label within the datasets, I create a text file that contains only the textual content. This file is used for textual analysis and contextual analysis purposes.

3. Analyzing Datasets:

This is the most critical phase of the study. The analysis phase adds value and insight into the content of the datasets. I present detailed investigations for the content of

each dataset by conducting statistical, textual, and contextual analysis, in addition to generating multiple graphs to visualize the content.

a. Statistical analysis

The statistical analysis includes finding frequencies of words, frequencies of stop words, statistical measurements for the lengths of the text based on the number of tokens, and statistical measurements for the lengths of the tokens based on the number of characters to analyze their relationships with offensive content. To extract the most frequently used words for each class accurately, I remove a list of stop words from the text. The stop words list includes the NLTK Arabic stop words list, and Albadi, Kurdi, and Mishra (2018)'s stop words list. Then, I search for the words that have the prefix "ال" to remove the prefix. I do not remove the prefix "ال" when it is used as a part of the word and not as a prefix, such as in the word "الله". A simple count of token frequencies is useful to compare among multiple classes; however, it does not provide rich information about each class separately. I use the web-based tool Voyant[16] to further analyze the text and identify the top five most distinctive words of each class. Stop words could help in defining the context of the posts. I conduct simple frequency analysis to generate the top stop words per class, as stop words that appear only in a particular class might be better to consider in the analysis as a regular word rather than as a stop word. I investigate the complexity of the text used in each class to check if there is any pattern or relationship between the complexity of the text

[16] https://voyant-tools.org/

used and the type of offensive content. I use two measures to support this analysis; the number of characters per token and the number of tokens per post.

b. Textual analysis

Before cleaning or filtering the data, I generate word cloud graphs for each label from each dataset using the textual files to give some intuition about the raw content of each class. Data in all datasets are extracted from user-generated content platforms that are usually written in unstructured format and using dialectal Arabic, which is not supported by most of the available textual analysis tools. Thus, I could not perform POS tagging to analyze the text based on their functional roles and investigate whether that could influence the offensive content.

c. Contextual analysis

I study the impact of context on offensive content. Context is defined in terms of text sentiment, the use of emojis, and the use of punctuations. To better understand the context of the samples, I use the Mazajak online tool[2] for Arabic sentiment analysis to predict the sentiments of tweets. Thus, each sample is classified as positive, negative, or neutral depending. on its content. Emoji is often used in online communication to reflect the emotion and express personality; thus, considering emojis adds value to understanding text. Punctuations provide a clue for the meaning of unfamiliar phrases and the context of the sentence. As a result of that, I analyze the use of punctuations and their effects on offensive content.

4. Summarizing and Synthesizing Results:

After reviewing the analysis section, I connect results across the datasets and summarize the overall findings. I add more insight into the findings by synthesizing the result with findings from previous studies and provide valuable design considerations for other researchers in the same domain of research.

Datasets Analysis Results

This section contains the results from dataset analysis in chronological order based on the publication date of each dataset. A total of nine datasets satisfy the selection criteria: Aljazeera.net Deleted Comments, Egyptian Tweets, YouTube Comments, Religious Hate Speech, Levantine Hate Speech and Abusive Language, Tunisian Hate Speech and Abusive Language, Multi-Platform Offensive Language Dataset, the Fourth Workshop on Open-Source Arabic Corpora and Corpora Processing Tools, and the Multi-Platform Hate Speech Dataset.

The Aljazeera.net Deleted Comments Dataset

The Aljazeera.net deleted comments datasets is developed by Mubarak, Darwish, and Magdy (2017). It includes a total of 31,692 comments consisting of 5,653 clean comments, 533 obscene comments, and 25,506 offensive comments. The total number of duplicate comments is 8; 2 clean comments, 1 obscene comment, and 5 offensive comments. Figure 20 shows class distribution.

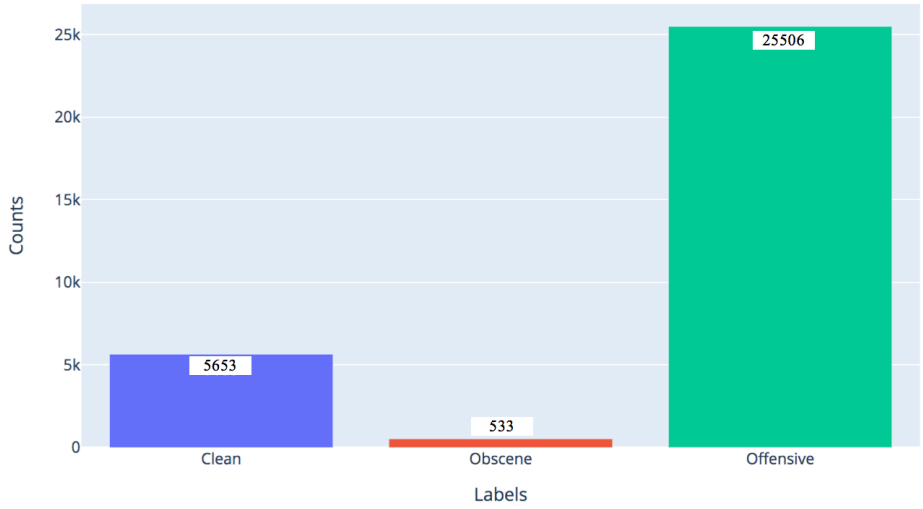


Figure 20 Class distribution for the Aljazeera dataset

The following is an example offensive comment:

أناديكم أناديكم أشد على أياديكم وأبوس الأرض تحت نعالكم .. تجار الدين من الروفض والايرانين يجب تطهيرهم كما فعل بهم صلاح الدين . بعدها سوف تتحرر الاراضي العربيه من الاحتلال الفارسي والاسرائيلي

Translation: I am calling you, I am calling you and hold your hands and kiss the land beneath your shoes.. The land needs to be cleaned from the Iranian and Shia as Salah Al-Deen did before; after that, the Arabic land will get free from the Persian and Israeli invasion.

Investigating text through the word cloud from Figure 21, it can be noticed that the most common particles differ among the three classes. For example, in clean comments, "هو / he" and "في /in" are the most frequent ones, in obscene comments, "يا /you" and "لا /no" are the most used ones, and in offensive comments, "كان /was" and "من / from" are more likely to be seen than other particles. From the word cloud figure, some

distinguishable words among the classes are "فيصل / Faisal" in clean comments, "كس / pussy" in obscene comments, and "سيف / sword" in offensive comments.

(a) Clean (b) Obscene

(d) Offensive

Figure 21 The word cloud of the Aljazeera dataset (a. clean, b. obscene, c. offensive)

In Figure 22 to Figure 24, the top 10 frequent tokens are shown for each class separately. In all classes, the words "الله / God" is the most frequent one. For both clean and offensive comments, the second most frequent word is "دوله / state" while for obscene comments, the second most frequent word is "ابن / son". The third most frequent word in clean comments is "جزيره / Jazeera", "كس/ pussy" in obscene comments, and "مصر / Egypt" in offensive comments.

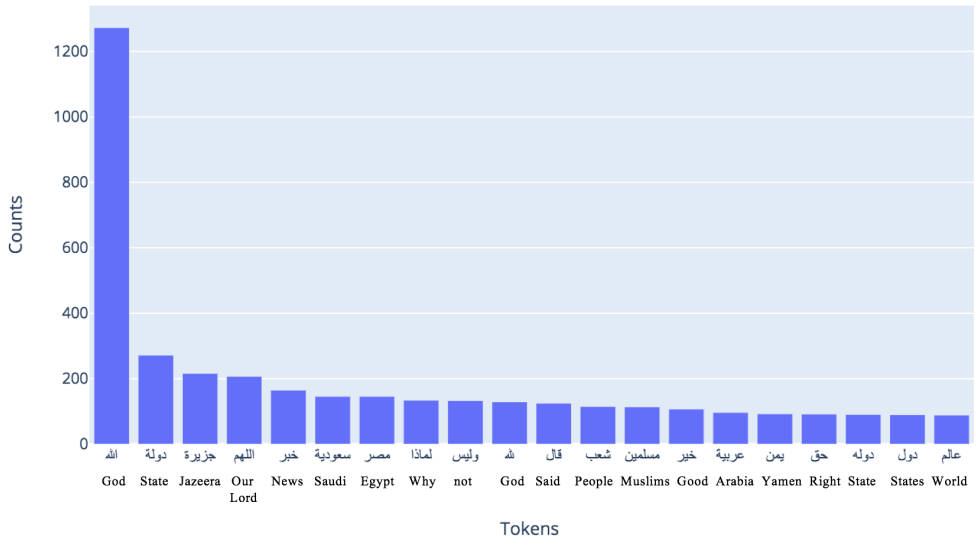


Figure 22 Most common tokens in the clean class in Aljazeera dataset

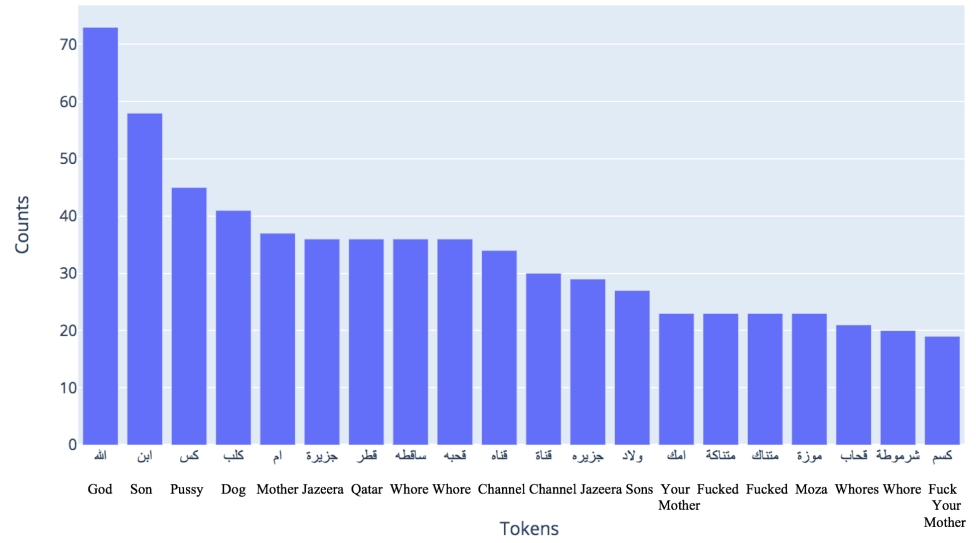


Figure 23 Most common tokens in the obscene class in Aljazeera dataset

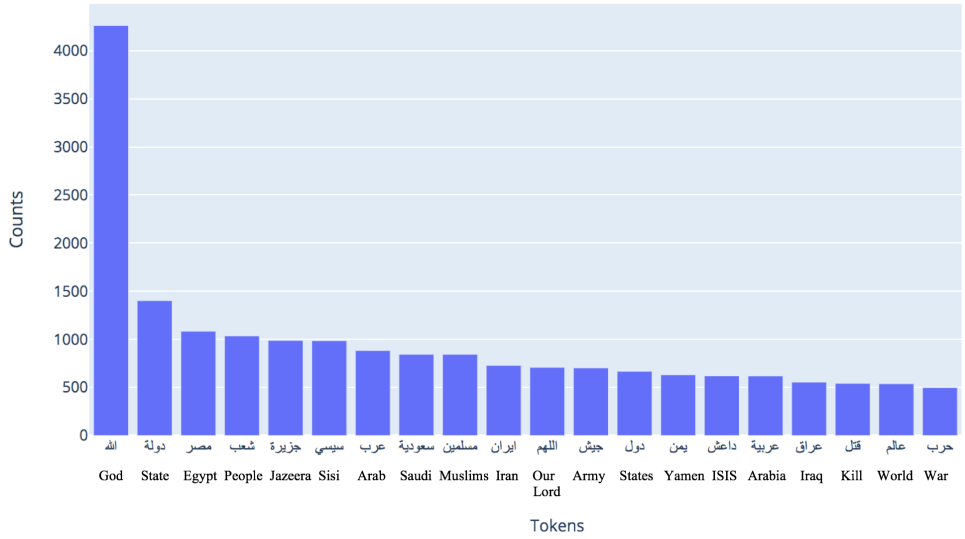


Figure 24 Most common tokens in the offensive class in Aljazeera dataset

Applying the web-based tool Voyant shows the following five distinctive words for each class:

1. Clean: الخبر / the news (130), قال / said (128), ووفقكم / bless you (38), عافاكم / make you healthy (36), خير / good (92).
2. Obscene: المتناك / fucked (20), كسم / fuck your mother (19), كس / pussy (44), المتناكة / fucked (16), القحبة / whore (13).
3. Offensive: قال / said (445), الاسد / Al-Asad (275), السوري / Syrian (271), تركيا Turkey (244), فلسطين / Palestine (227).

Figure 25 and Figure 26 compare the results of the statistical analysis for the length of comments and the length of tokens based on the classes. In all cases, offensive comments are the longest followed by obscene comments and clean comments. Clean comments are the only category that has some outliers for comments length.

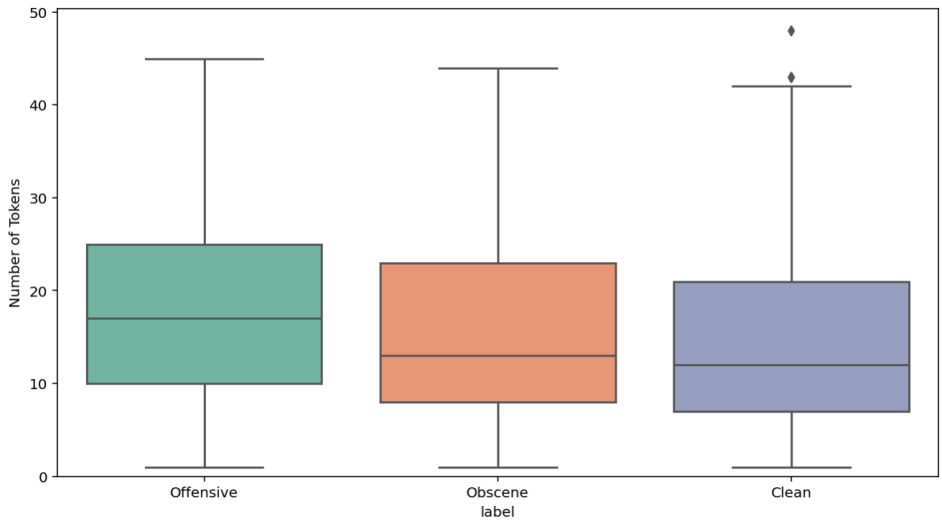


Figure 25 Statistics of each label in Aljazeera dataset based on the number of tokens per comment

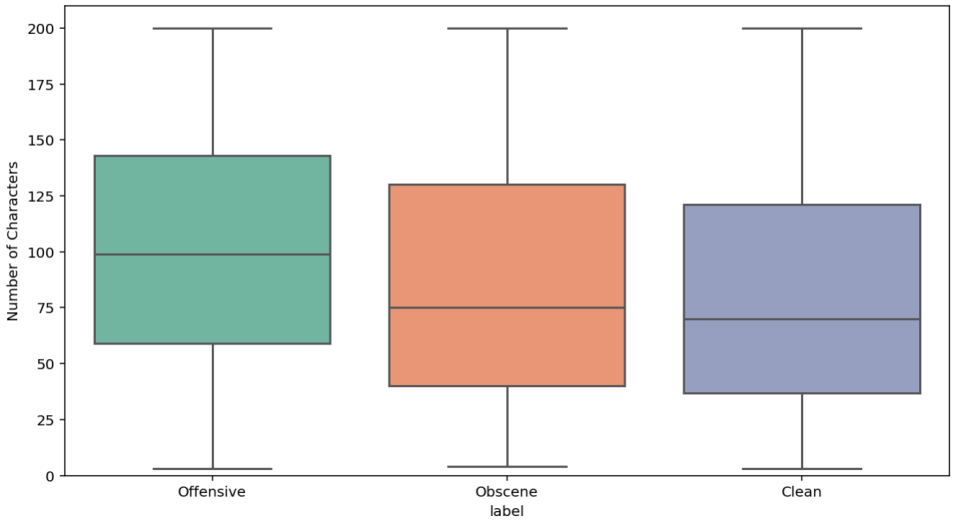


Figure 26 Statistics of each label in the Aljazeera dataset based on the number of characters per token

Investigating the use of stop words among the classes from Figure 27 to Figure 29, shows a very similar pattern among all classes. For example, "من / from" is the top

one among all, followed by "و / and" in both clean and offensive classes and "يا / you" in obscene class, and "في / in" is the third most frequent stop word among all classes.

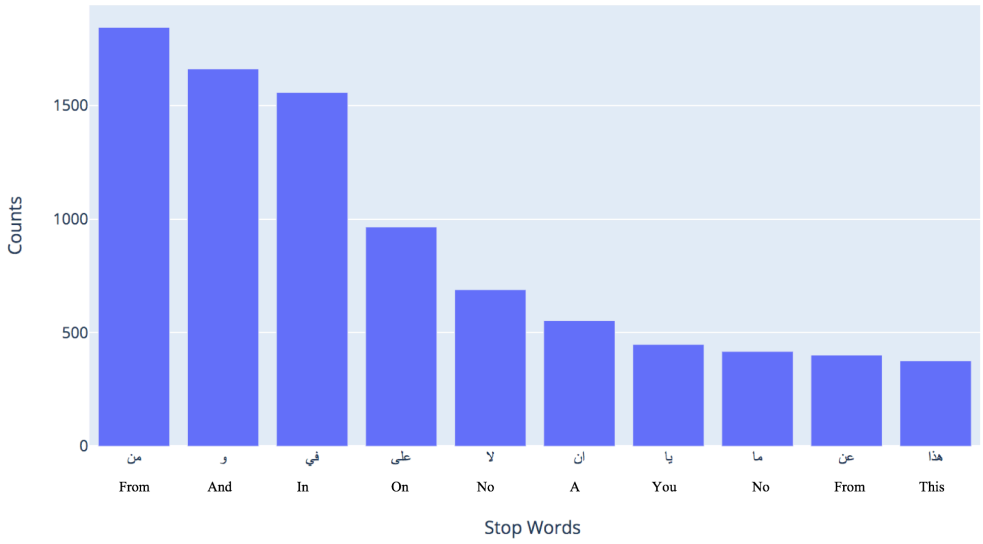


Figure 27 Most common stop words in clean class from the Aljazeera dataset

Figure 28 Most common stop words in obscene class from the Aljazeera dataset

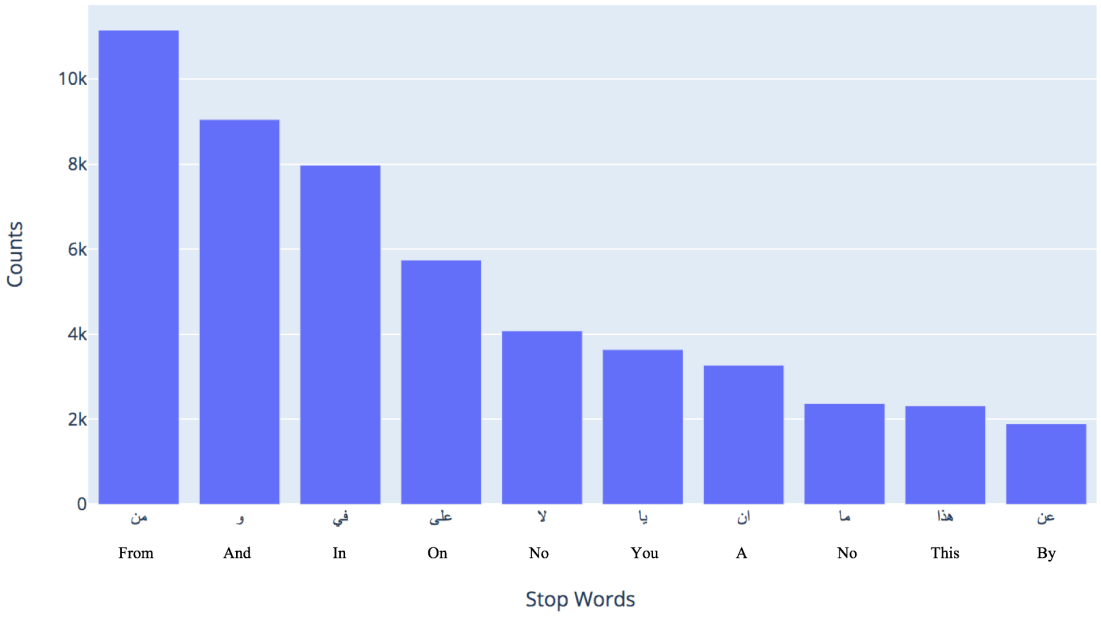


Figure 29 Most common stop words in offensive class from the Aljazeera dataset

Figure 30 is a bar chart for the sentiment analysis results. Obscene comments are all labeled negatively by the Mazajak online tool, while clean and offensive comments have mixed sentiments; mostly negative followed by neutral then positive.

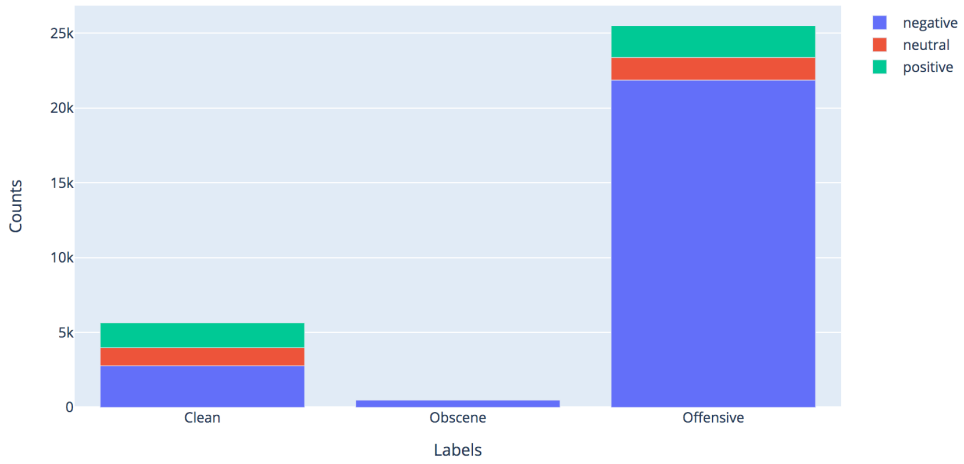


Figure 30 Sentiment analysis based on labels for Aljazeera dataset

The provided comments do not have any emojis, so I could not analyze the use of emojis among the classes. Figure 31 to Figure 33 show frequencies of the top ten punctuation marks for each class. As can be seen from the figures, the three most frequent punctuation marks are the same for all classes; “.”, “!”, and “؟”.

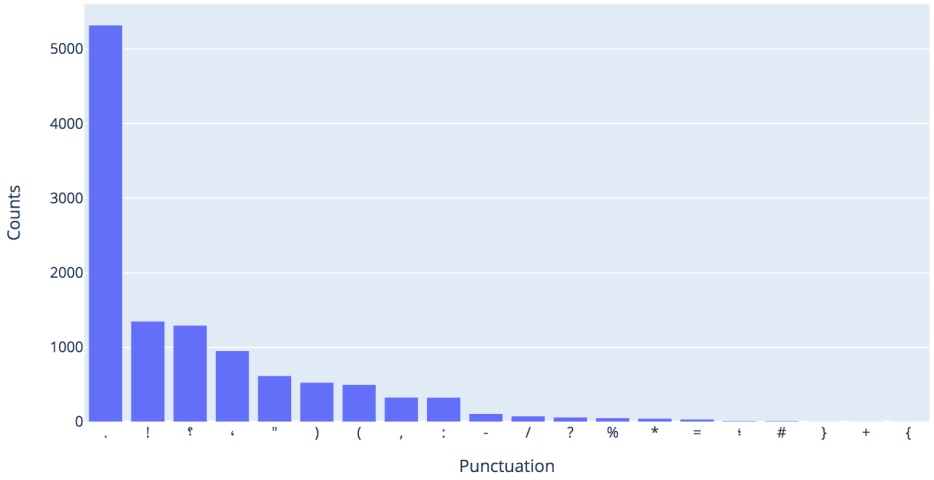


Figure 31 Most common punctuation in the clean class in Aljazeera dataset

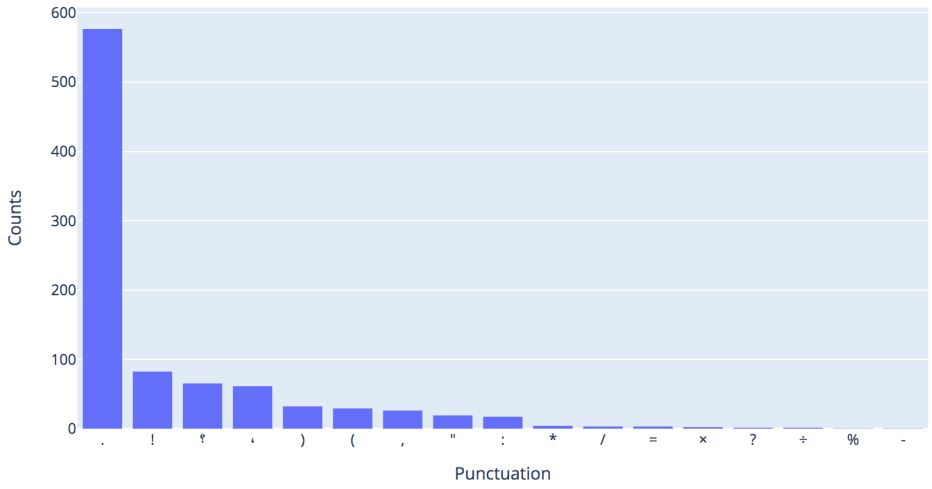


Figure 32 Most common punctuation in the obscene class in Aljazeera dataset

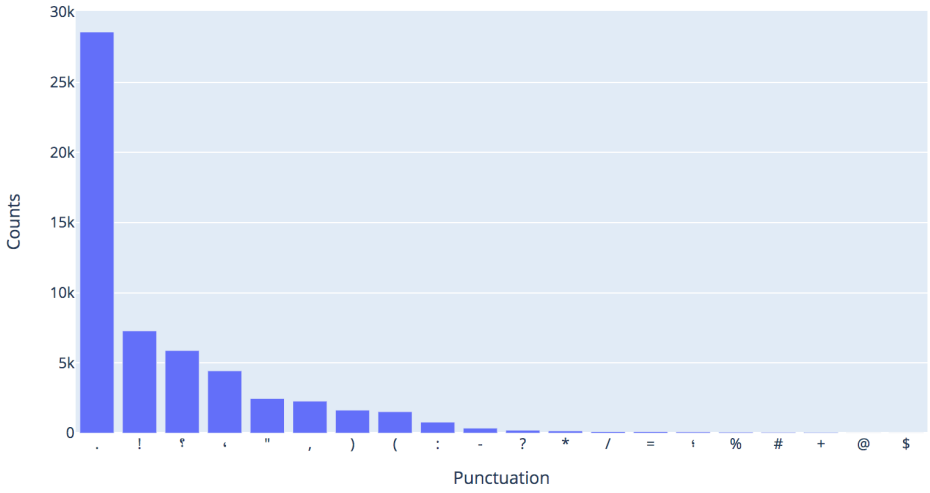


Figure 33 Most common punctuation in the offensive class in Aljazeera dataset

Similar analysis for the rest of the datasets has been conducted and a summary of the result is in

Table 12 and Table 13.

Egyptian Tweets Dataset

The Egyptian tweets dataset was developed by the same researchers who compiled the Aljazeera deleted comments dataset, and it has the same labeling structure (Mubarak, Darwish, & Magdy, 2017). The total number of tweets is 1,100; 453 clean, 203 obscene, and 444 offensive. The total number of duplicate tweets was two.

Religious Hate Speech Dataset

Albadi, Kurdi, and Mishra (2018) publish the Religious Hate Speech Dataset, which consists of 6,137 Arabic tweets; 2,762 hate, 3,375 not hate. The dataset has multiple duplicates of eight original tweets.

YouTube Comment Dataset

Alakrot, Murray, and Nikolov (2018) extract a comments dataset from a set of controversial YouTube channels that contains a total of 15,050 comments. Classes are imbalanced with 9,237 not offensive comments and 5,813 offensive comments. The dataset has two comments that have been repeated 11 times.

Levantine Twitter Dataset for Hate Speech and Abusive Language (L-HSAB)

The L-HSAB dataset is a dialectal dataset that contains 5,846 Levantine tweets (Mulki et al., 2019). It has three classes: hate = 468 tweets, abusive = 1,728 tweets, and normal = 3,650 tweets. Multiple identical repetitions of tweets were found.

The Tunisian Hate and Abusive speech dataset (T-HSAB)

Haddad et al. (2019) develop T-HSAB dataset, which has a total of 6,075 comments; 3,834 normal, 1,127 abusive, and 1,078 hate. Duplicate comments were removed.

The Multi-Platform Offensive Language Dataset (MPOLD)

Chowdhury et al. (2020) develop the MPOLD dataset, which consists of 4,000 comments; 3,325 are not offensive comments and 675 are offensive. Offensive comments are further classified as vulgar, hate, and other. For the purposes of this research, I just focus on the first level label hierarchy.

The Fourth Workshop on Open-Source Arabic Corpora and Corpora Processing Tools Dataset (OSACT)

The OSACT dataset is released by Mubarak et al. (2020). Each tweet has two labels; the first label used to classify the tweet into offensive or not offensive and the second one to classify it into hate speech or not hate speech. The total number of tweets is 10,000; 1,900 are offensive tweets and out of these offensive tweets, only 500 are hate speech.

The Multi-Platform Hate Speech Dataset

Omar, Mahmoud, and Abd El-Hafeez (2020) release the first multi-platform dataset for Arabic hate speech detection. Comments were collected from four social media platforms; Facebook, Twitter, YouTube, and Instagram. The dataset is balanced with 10,000 hate comments and 10,000 not hate comments.

Summarizing Results

The following table summarizes some features of the datasets that could be helpful to measure the quality of the datasets. The main quality-related attributes include the size of the dataset and whether the classes are close in size, the number of the samples that have been included in the dataset several times, the rounded average length of

samples (comments) based on the number of tokens, the rounded average length of tokens based on the number of characters, the inclusion of emojis and punctuation, and whether the content is pre-processed for privacy and security reasons.

Table 12 Quality features of the datasets

Dataset	Size per label	No. of Duplicates	Avg. length of comment (character)	Emojis	Punctuation	Anonymized
The Aljazeera.net Deleted Comments	31,692 (5,653 clean, 533 obscene, 25,506 offensive)	7 original comments	15 tokens (80 characters)	Not availabl e	Available	Not available
Egyptian Tweets	1,100 (453 clean, 203 obscene, and 444 offensive)	2 original tweets	13 tokens (79 characters)	Availab le	Available	Not available
Religious Hate Speech	6,137 (2,762 hate and 3,375 not hate)	8 original tweets	22 tokens (140 characters)	Availab le	Available	Not available
YouTube Comments	15,050 (9,237 not offensive and 5,813 offensive)	2 original comments	10 tokens (100 characters)	Availab le	Available	Not available
L-HSAB	5,846 (468 hate, 1,728 abusive, and 3,650 normal)	No duplicates	10 tokens (70 characters)	Availab le	Available	Not available
T-HSAB	6,075 (3,834 normal, 1,127 abusive, and 1,078 hate)	No duplicates	9 tokens (80 characters)	Not availabl e	Not available	Not available
MPOLD	4,000 (3,325 not offensive and 675 offensive)	No duplicates	10 tokens (80 characters)	Availab le	Available	Available
OSACT	10,000 (8,100 not offensive and 1,900 offensive)	3 original tweets	19 tokens (95 characters)	Availab le	Available	Available
Multi-Platform Hate Speech	20,000 (10,000 hate and 10,000 not hate)	2 original comments	30 tokens (150 characters)	Not availabl e	Available	Not available

The content of Table 2 summarizes the main findings for each dataset based on the class. This summarization supports the comparison of each type of offensive content (e.g., hate speech, offensive, obscene) among the entire Arabic offensive resources, which can provide a holistic picture of the available resources.

Table 13 Label-specific attributes summary

Dataset	Label	Top three tokens	Top three distinctive words	Top three stop words	Major sentiment	Top three emoji	Top three punctuation
The Aljazeera.net Deleted Comments	Clean	الله /God, دولة /state, جزيره /Jazeera	الخبر /the news, قال /said, ووفقكم /bless you	من /from, و /and, في /in	Negative	Not available	., !, ؟
	Obscene	الله /God, ابن /son, كس /pussy	المتناك / fucked, كسم / fuck your mother, كس / pussy	من /from, يا / you, في /in	Negative	Not available	., !, ؟
	Offensive	الله /God, دولة /state, مصر /Egypt	قال /said, الاسد /Al-Asad, السوري /Syrian	من /from, و /and, في /in	Negative	Not available	., !, ؟
Egyptian Tweets	Clean	الله /God, مصر / Egypt, شعب /people	الحل /the solution, الرئيس /the president, علاقة / relationship	من /from, في /in, و /and	Negative	😢, 😂, 😭	., “, #
	Obscene	امك /your mother, عرص /bad behaved, كلب /dog	العرص /the bastard, امك /your mother, عرص /bastard	يا / you, من /from, في /in	Negative	😂, ✌, 🐶	., “,!
	Offensive	مصر / Egypt, الله /God, منك/ from you	البرادعى /El-Baradei, بطل /hero, اهبل /stupid	يا / you, من /from, في /in	Negative	😂, 😢, ✋	., “,؟
Religious Hate Speech	Not hate	الله /God, يهود /Jews, شيعة /Shia	فطرة/ primitiveness, الإخلاص/the sincerity, إبراهيم/ Ibrahim	من /from, في /in, على /on	Negative	😂, 🌹, 🖤	., /, #
	Hate	الله /God, لهم /for them, مسلمين /Muslims	اللعنة /curse, العهر /immorality, متحدون /united	من /from, في /in, على /on	Negative	😂,🇮🇱, 👇	., /, #
YouTube Comments	Not offensive	كاظم/ Kadhim, الله/ God, احلام/ Ahlam	بقناتي / in my channel, مها / Maha, يهديك / give you	من /from,و / and, في /in	Negative	😂, 🖤, 😍	.,؟ , +
	Offensive	الله /God, كاظم/ Kadhim, احلام/ Ahlam	زباله /trash, زبالة /trash, خرة /shit	من /from, يا / you, و / and	Negative	😂, 👎, 😒	., ؟, ،
L-HSAB	Normal	جبران باسيل/ Gebran Bassil, وزير/ minister, الله/ God	القمة /the top, الخارجية /the external, الوطني /the national	من /from, ما / what, في /in	Negative	🖤, 😎, ♀	،, ؟, ؛

	Abusive	كول هوا/ eat air, كلب/ dog	خرا /shit, واطي /cheap man, شرفك /your honor	يا / you, , ما / what, في /in	Negative	😂, 😎, 🤷‍♂️	؟, ، , ؛
	Hate	عالمطبخ /to the kitchen, البنات /girls, ولاك /guys	عالمطبخ /to the kitchen, البنات /girls, ولاك /guys	يا / you, من /from, و/ and	Negative	😂, 😎, ♂	؟, ، , !
T-HSAB	Normal	تونس / Tunisia, ربي“ /my lord, لطفي /Lutfy	عواطف /emotions, الخطاب / Al-Khattab, صلاح / Salah	ه /ha ,أكثر / more, هكذا /this is	Negative	Not available	Not available
	Abusive	قحبة /whore, تونس / Tunisia, نيك /fuck	نيك /fuck, قحبة /whore, القحبة / the whore	ه /ha ,أكثر / more, بكم / by you	Negative	Not available	Not available
	Hate	تونس / Tunisia, شعب /people, دين /religion	القحبة / the whore, ودين/ religion, قحبة /whore	ه /ha ,أكثر / more, الآن /now	Negative	Not available	Not available
MPOLD	Not offensive	@User.IDX, الله/ God, قناة /channel	شكرا / thanks, برنامج /program, افضل/ better	من/ from, في/ in, و / and	Negative	😂, 🖤, 👍	., @, ،
	Offensive	@User.IDX, الله/ God, قناة /channel	الخنزيره /pig, الحمدين /the two Hamad, قذر /dirty	من/ from, في/ in, و / and	Negative	😂, 👇, 🤣	., @, !
OSACT	Not offensive	@USER, RT, URL	قيوم /everlasting, مجيب /answerer, رحمن /merciful	يا / you, من /from, و/ and	Positive	😂, 🖤, 💙	<, >, .
	Offensive	@USER, الله/ God, URL	فسواي /shit, كلووووب /small dog, كلبه /female dog	يا / you, و/ and,من /from	Negative	😂, 🤣, 💙	@, ., <
Multi-Platform Hate Speech	Not hate	اللهم /our God, الله / God, تقنية /technology	تقنية /technology, أندرويد /Android, الطبيب /doctor	من /from, في/ in, على /on	Neutral	Not available	., ، , :
	Hate	الله / God, ام / mother, كس / pussy	كس /pussy, كسمك /your mother's pussy, عرص /bad behaved	من /from, يا / you, و/ and	Negative	Not available	., #, ،

Synthesizing Results

The problem of online offensive language is a very complex one. The analysis shows that most surveyed datasets have an average length of input text less than or equal to 15 tokens, making it difficult to accurately identify the context and differentiate between offensive content and other similar content such as sarcasm content. This problem gets more complicated when the input text does not include non-textual

elements, such as emojis or pictures, that might otherwise add more insight into the textual content. In addition, the majority of the surveyed datasets are imbalanced with a very small percentage of offensive content, which makes it very difficult to depend on one dataset to develop an offensive language detection system with sufficient training instances and obtain accurate results.

Offensive words differ among the datasets regardless of the offensive text type. This variation could be a result of Arabic dialects and Arabic sub-cultures represented in the datasets. Overall, Arabic users of online social media are commonly using the face with tears of joy emoji "😂", in their conversations as it appears as the top frequent emoji in most surveyed datasets for offensive and not offensive samples. The use of punctuation does not demonstrate any specific patterns among the datasets. Stop word analysis highlights the relationship of "يا / you" with offensive content as it appears among the most frequent stop words in multiple offensive types from multiple datasets. Some of the most frequent tokens are part of a name of famous figures; for example, "كاظم/ Kadhim", and "احلام/ Ahlam" from YouTube Comments dataset; thus, it would be better to consider studying the relationship between names of famous persons and offensive content.

Design Considerations

The available Arabic offensive language datasets that I discuss in this paper cover several offensive contents from different platforms with different Arabic dialects that can provide valuable insights into the problem of online offensive content. Accordingly, researchers in this domain of research need to develop advanced methods to extract

collective knowledge from all available offensive language datasets in the Arabic language in order to address the problem from several dimensions with a holistic view.

Chapter Summary

In this chapter, I investigate the content of nine Arabic offensive language datasets to provide an in-depth analysis of their content. This investigation aims to guide researchers in Arabic offensive language detection in selecting the appropriate dataset based on content and in creating new Arabic offensive language resources to support and complement the available ones. Results demonstrate the limited content of the surveyed datasets in terms of sample sizes and the lengths of the samples. Results also show variations in the offensive content among the datasets. Thus, it is very important to consider developing an innovative method to extract valuable insights about Arabic offensive content from the available datasets collectively, and to apply that knowledge for an automatic Arabic offensive language detection system.

WORD REPRESENTATION- THE BERT MODEL

" يَا أَيُّهَا الَّذِينَ آمَنُوا لَا يَسْخَرْ قَومٌ مِّن قَوْمٍ عَسَى أَن يَكُونُوا خَيْرًا مِّنْهُمْ وَلَا نِسَاء مِّن نِّسَاء عَسَى أَن يَكُنَّ خَيْرًا مِّنْهُنَّ وَلَا تَلْمِزُوا أَنفُسَكُمْ وَلَا تَنَابَزُوا بِالْأَلْقَابِ"

"O you who believe, no men should ever scoff at other men. May be, the latter are better than the former. Nor should women (ever scoff) at other women. May be, the latter women are better than the former ones. And do not find fault with one another, nor call one another with bad nicknames."

-Holy Qur'an, chapter Al-Hujurat (49), verse 11

This chapter provides the foundation to the next chapters by examining the AraBERT model to arrive to the best settings for experimental studies. Accordingly, this chapter focuses on explaining the BERT models and the concept of transfer learning. It also includes a short experimental study to tune AraBERT hyperparameters.

BERT stands for Bidirectional Encoder Representations from Transformers (Devlin et al., 2018). It is an innovative language model that presents SOTA results in multiple NLP tasks, such as question answering and language inference. BERT applies pre-trained language representations to down-stream tasks through a fine-tuning approach. This approach is also called transfer learning, in which the pre-trained language representations are developed using a neural network model on a known task, and then fine-tuning is performed to use the same model for a new purpose-specific task. The main feature that distinguishes BERT from the other language modeling techniques

is the use of a bidirectional language model rather than a unidirectional language model during the fine-tuning process. This bidirectional learning technique consists of a Masked Language Model (MLM) with a pre-training objective that randomly masks some of the tokens from the input to predict the original masked token based only on its context (Devlin et al., 2019).

This chapter helps in answering the following questions:

- What is transfer learning? And how is it applied in NLP?
- Why is BERT model very powerful in NLP tasks?
- Which hyperparameters work the best for the AraBERT model when applied to Arabic offensive language detection?

The chapter is organized as follows: a general background about transfer learning in NLP is reviewed at the first section, then more in-depth detail about BERT model architecture is given. The third section is dedicated to explaining how BERT constructs contextual word embeddings. The AraBERT model is discussed in the fourth section. Section five includes an experimental study to tune AraBERT hyperparameters for Arabic offensive language detection. The chapter ends with a summary that contains the main lessons from this chapter and connects it to the rest of this thesis.

Transfer Learning in NLP

It is critical to understand the main concepts behind transfer learning in order to understand the BERT model because it is developed based on transfer learning theory. Previous studies report SOTA performance results by applying transfer learning for many

supervised NLP tasks (e.g., classification, information extraction, etc.) (Devlin et al., 2018). Most NLP tasks share common linguistic knowledge, such as linguistic representations and structural similarities in which transfer learning adds value by allowing tasks to inform each other (e.g., syntax and semantics). In addition, labeled datasets are rare and costly; thus, transfer learning can help make the most out of using as much supervision as available (Wolf, 2019). Accordingly, adopting transfer learning for NLP tasks saves time and effort by enabling language processing models to be initialized with existing prior knowledge (Azunre, 2021).

Ruder (2019) proposes a taxonomy for transfer learning in NLP in his dissertation "Neural Transfer Learning for Natural Language Processing", as shown in Figure 34.

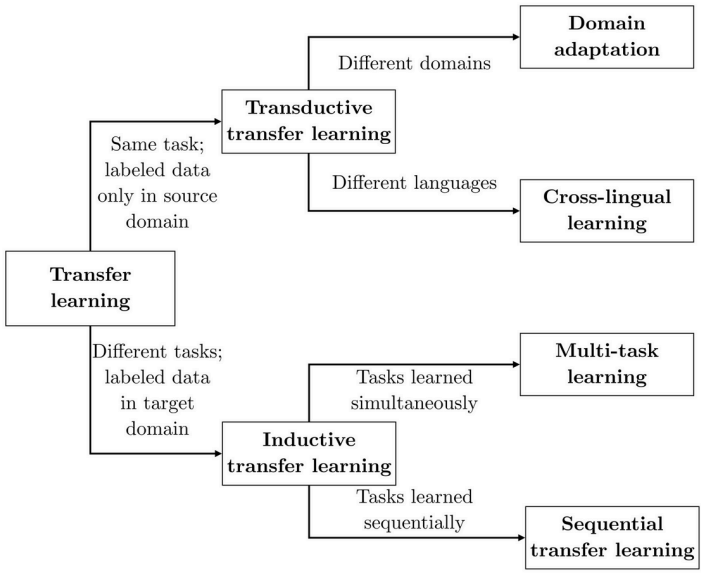


Figure 34 taxonomy for transfer learning in NLP (Ruder, 2019, p.5)

The first branch of transfer learning is called transductive transfer learning, which combines datasets for the same NLP task with labeled data only in the source domain dataset. There are two types of learning within transductive transfer learning; the first one is called domain adaptation. In domain adaptation, two different domain datasets are used for the same task. For example, a system for question answering on news can be applied to a corpus of customer surveys. Transductive transfer learning can be applied across different languages, such as Arabic and English, so that knowledge learned from one language can serve the task for the other one. This type of transfer learning is discussed in more detail in chapter 8.

The second branch of transfer learning is applied to different NLP tasks with labeled data in the target domain dataset, and is called inductive transfer learning. The first type of inductive transfer learning is multi-task learning, which consists of having multiple different tasks running simultaneously. The second type is called sequential transfer learning that also has multiple tasks, but they are applied in sequential order, for example, learning word representation first (called pre-training) and then applying to the target task such as offensive language detection (called adaptation). The BERT model is developed based on the sequential transfer learning approach. These four types of transfer learning are not discrete, they could overlap.

The BERT Architecture

There are two main components of BERT; model and training. The model includes the pre-training architecture and the fine-tuning (adaptation) architecture. In general, all architectures depend on transformers, i.e., generic models based on attention

mechanism (encoder and decoder). Figure 35 shows the architecture of the transformer layer. Model architectures vary in size from simple architecture such as in XLNeT to very large architecture such as in XLM. For example, BERT-Base has 12 Transformer layers (blocks), with 768 hidden layers, 12 bidirectional self-attention heads, and a total of 110M parameters, while BERT-Large has 24 Transformer layers (blocks), 1,024 hidden layers, 16 bidirectional self-attention heads, and a total of 340M parameters.

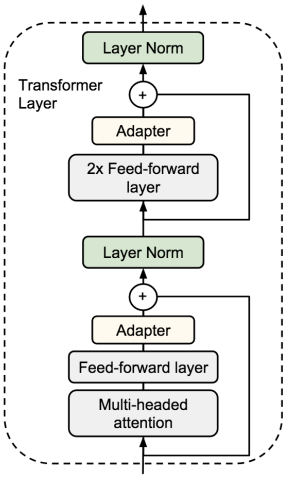


Figure 35 Transformer layer (Houlsby et al., 2019, p.3)

The training component of BERT consists of pre-training and fine-tuning phases. A long stream of continuous text is used to help the model learn semantic and syntactic long-term dependencies in the text that is used during pre-training. Considering the quality and the size of the data is very important during training.

The BERT Model as a Contextual Word Vector

The contextual word vector creates one vector for all contexts instead of capturing each word separately. Unlike most of the previous contextual word embeddings (e.g., GloVe), BERT applies WordPiece embeddings. WordPiece is an algorithm for sub-word segmentation, in which the vocabulary is initialized with individual characters from the text. After that, BERT iteratively adds the most frequent combinations of characters in the vocabulary (Wu et al., 2016). Each segment of the word (WordPiece) is assigned to a numeric ID. Furthermore, BERT contextual embedding outperforms previous contextual embeddings because BERT pre-trains both sentence and contextual word representations, using MLM and next sentence prediction with multiple layers of transformers. As can be seen from Figure 35, the multi-head bidirectional self-attention mechanism in the Transformer layers increase the understanding of language during the pre-training phase of the language model because it considers the contextual relationship of each word (target word) to the others before and after it on the same time from the entire corpus using a summation of the corresponding token, segment, and position embeddings. A special classification token [CLS] is used for the first token of every sequence. Another special token is [SEP], which is used to separate sentence pairs. Thus, the positional embedding provides the model with relative positioning knowledge (Devlin et al., 2018). Figure 36 is taken from the original BERT paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" to illustrate how the encoding is performed in BERT.

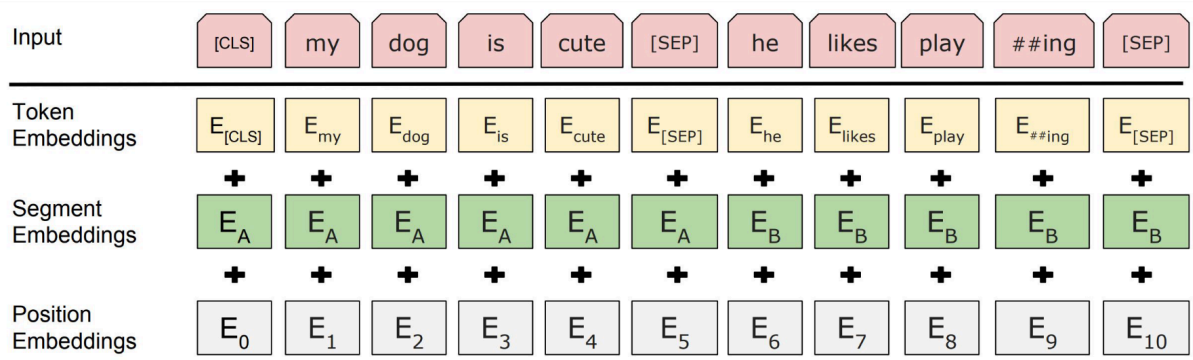


Figure 36 BERT input representation (Devlin et al., 2018, p.5)

This encoding process is conducted during the model training phase, which allows BERT to construct the linguistic knowledge that makes it a powerful language model. Scaling-up the model by increasing the size of the corpus used during training phase helps increase the model's knowledge. The knowledge that is acquired by the pre-trained BERT model is transferred through the fine-tuning process to serve other tasks such as, in our case, offensive language detection.

The AraBERT Model

The AraBERT model is a monolingual Arabic BERT model. It has various versions with variations in the model architecture and training corpus; AraBERTv1-base model is 543MB in size with 136M parameters and pre-trained using 77M sentences that are 23GB in size and 2.7B words, while AraBERTv2-large is 1.38G size with 371M parameters and pre-trained with 200M sentences that are 77GB in size and 8.6B words. In this dissertation, I focus on BERT-base-AraBERT (AraBERTv1-base) and refer to it as AraBERT. More details about AraBERT are discussed in Chapter 4. AraBERT is

trained mostly on MSA Arabic News outlets, the following are the sources including in its training corpus:

1. Arabic Wikipedia database dump[17]
2. 1.5B words Arabic Corpus[18] (sources include newspapers, books, and research papers)
3. Open-Source International Arabic News Corpus (OSIAN)[19]
4. Assafir Lebanese news articles[20]

AraBERT Fine-Tuning Hyperparameters Optimization Method

The selection of the AraBERT's hyperparameters for the next chapters is based on the findings from the hyperparameters optimization method conducted in this chapter. I utilize all nine datasets discussed in Chapter 5. I start by creating a training set and testing set using 80%-20% random splits by applying Scikit-Learn Python library to ensure preserving the same distributions of labels. I binarize all labels to have only offensive and not offensive classes by converting the different types of offensive classes into one offensive class. Then, I concatenate the training portions of all datasets into one corpus to be used for hyperparameters tuning. I apply 5-fold cross validation on the concatenated training corpus to arrive at the most efficient hyperapameters. While there are multiple hyperparameters in BERT, I select the most sensitive ones based on the recommendations from the original BERT paper (Devlin et al., 2018). AraBERT authors do not provide

[17] https://archive.org/details/arwiki-20190201
[18] https://arxiv.org/pdf/1611.04033.pdf
[19] https://www.aclweb.org/anthology/W19-4619/
[20] http://m.assafir.com/Channel/50/English/TopMenu

recommendations on the hyperparameters. According to Devlin et al. (2019), best hyperparameter values are task-specific, but the following values are proven to work well for all tasks:

- Batch size: 16, 32
- Learning rate (Adam): 5e-5, 3e-5, 2e-5
- Number of epochs: 2, 3, 4

Datasets

The same datasets explored in Chapter 5 will be used on the remaining chapters based on each experiment's settings. Thus, I use the training portion from all datasets to tune the hyperparameters in this chapter and apply the optimized hyperparameters on the next chapters. To do so, I apply the same preprocessing steps conducted on remaining chapters, including 80-20% train-test random split and label binarization. The table below shows the datasets' distributions for the training and testing sets, as well as datasets dialect and source's platform.

Table 14 Datasets distributions

Dataset	Platform	Dialect	Training Set	Testing Set
The Aljazeera.net Deleted Comments	Aljazeera News	Multi-dialect	25,353 comments (not offensive = 4,528, offensive = 20,825)	6,339 comments (not offensive = 1,125, offensive = 5,214)
Egyptian Tweets	Twitter	Egyptian	880 tweets (not offensive = 353, offensive = 527)	220 tweets (not offensive = 100, offensive = 120)
Religious Hate Speech Tweets	Twitter	Multi-dialect	12,040 comments (not offensive = 7,379, offensive = 4,661)	3,010 comments (not offensive = 1,858, offensive = 1,152)
YouTube Comments	YouTube	Multi-dialect	4,909 tweets (not offensive = 2,726, offensive = 2,183)	1,228 tweets (not offensive = 649, offensive = 579)
L-HSAB Tweets	Twitter	Levantine	4,819 samples (not offensive = 3,068, offensive = 1,751)	1,205 samples (not offensive = 752, offensive = 453)
T-HSAB Tweets	Twitter	Tunisian	4,676 tweets (not offensive = 2,919, offensive = 1,757)	1,170 tweets (not offensive = 731, offensive = 439)
MPOLD	Multi-platform	Multi-dialect	3,200 samples (not offensive = 2,660, offensive = 540)	800 samples (not offensive = 665, offensive = 135)
OSACT	Twitter	Multi-dialect	8,000 tweets (not offensive = 6,403, offensive = 1,597)	2,000 tweets (not offensive = 1,606, offensive = 394)
Multi-Platform Hate Speech	Multi-platform	Multi-dialect	16,004 samples (not offensive = 7,973, offensive = 8,031)	4,001 samples (not offensive = 2,027, offensive = 1,974)

Classification Model

I develop the same classification model that is used in Chapter 6 and the next chapters. AraBERT model from HuggingFace[21] library is used with a simple FFNN layer to build the classifier. The experiment was developed in Python using the PyTorch-Transformers library, and evaluation metrics were developed using the Scikit-Learn Python library. Google Colab Pro was used to conduct the experiment.

Results

Table 15 shows the resulting average 5-fold cross-validation results for the combined training sets. As can be noticed from Table 15, I do not find any notable differences among the different values of the learning rate, batch size, and number of epochs. This finding is in line with Devlin et al. (2019)'s finding for fine-tuning to large datasets, i.e., 100k labeled training samples, which demonstrate lower sensitivity towards hyperparameters' values than small datasets. Thus, I select the smallest values of all hyperparameters as that can support faster processing time and more convenient with Google Colab Pro available processing power.

[21] https://huggingface.co/

Table 15 Hyperparameter optimization results

Average Macro-F1 over 5-fold cross-validation		Learning Rate (Batch size)					
		2e-5 (16)	2e-5 (32)	3e-5 (16)	3e-5 (32)	5e-5 (16)	5e-5 (32)
Number of Epochs	2	0.88	0.88	0.88	0.87	0.88	0.88
	3	0.88	0.88	0.88	0.89	0.89	0.88
	4	0.89	0.89	0.88	0.88	0.88	0.88

Chapter Summary

This chapter reviews fundamental concepts in transfer learning for NLP, including transfer learning types and the BERT model. It also provides the foundations for the next chapters by introducing the datasets and tuning the hyperparameters. Results from hyperparameter optimization demonstrates no significant impact from changes in the learning rate, batch size, and number of epochs.

PRE-PROCESSING ARABIC TEXT

Imām Muhammad ibn ʻAlī al-Baqir (peace and mercy be upon them) said:
"إنَّ اللّه يُبغِضُ الفاحِشَ المُتَفَحِّشَ."
“Allah (God) hates the user of obscene language and the one who is shameless with it”

- Al-Kafi, volume 2, number 324; Mizan ul Hikmah, page 646

Pre-processing is one of the critical phases in most text classification pipelines. This chapter aims to investigate the impact of pre-processing on automatic offensive language detection in Arabic.

Pre-processing normalizes text in order to facilitate downstream processing. This normalization needs to preserve the original meaning of the text. Earlier studies investigating the impact of pre-processing on the text classification report different results for different pre-processing techniques and in different datasets (HaCohen-Kerner et al., 2020; Woo, Kim, & Lee, 2020). This finding implies the significance of conducting more analytical studies to understand better the behavior of pre-processing techniques on the text classification tasks. Moreover, very few researchers from the Arabic NLP field perform similar studies to investigate how preprocessing Arabic text can affect the performance of NLP systems (Duwairi and El-Orfali, 2014; Saad, 2010). While case lowering is one of the most widely used pre-processing techniques for English letters,

Arabic letters do not have lower and upper case. The Arabic language has unique characteristics that need to be considered during pre-processing.

The main goal of this chapter is to provide in-depth investigation and evaluation of the effects of intensive pre-processing on automatic offensive language detection in Arabic. I explore seven pre-processing techniques: conversion of emoticons and emojis to text, hashtag segmentation, normalization of different forms of Arabic letters, normalization of selected nouns from dialectal Arabic to MSA, conversion of selected hyponyms to hypernyms, and basic cleaning such as removing numbers, kashida, diacritics, and HTML tags. I evaluate multiple classifiers on each pre-processing step separately on two datasets. The classifiers include: (1) traditional machine learning classifiers; SVM and LR, (2) ensemble machine learning classifiers; Bagging and Random Forest, (3) ANN classifiers; RNN and LSTM, and (4) BERT-based models; AraBERT and Arabic-BERT.

The scope of this chapter covers the first hypothesis of the dissertation; *"H1. applying a contextual-aware preprocessing approach in cleaning and normalizing content of tweets increases the performance of the classifier"*; by addressing the following questions:

- What are the most significant preprocessing techniques for Arabic offensive language detection?
- How does preprocessing affect traditional machine learning classifiers and the more advanced classifiers?

- Do the results for the advanced classifiers imply the need for a change in the traditional text classification pipeline?

The following section starts with a related work section. Then, I discuss the methodology and the settings of the experiments, which include dataset description, preprocessing techniques, feature engineering, and classification models. The results of the experiments are presented after the methodology section. The chapter ends with a summary that presents the findings.

Related Work

There are few studies that focus on analyzing the effects of preprocessing Arabic text for text classification tasks. The pioneering study in investigating Arabic preprocessing text is delivered by Saad in 2010. Saad (2010) compares the effects of Arabic text pre-processing techniques on the performance of different traditional machine learning classifiers, including: Decision Tree, kNN, SVM, NB, and its variations; MNB, Complement NB (CNB), and Discriminative Multinomial Naïve Bayes (DMNB). Preprocessing techniques that were applied include tokenizing strings to words, normalizing tokenized words, removing stop words, different term weighting schemes, and Arabic morphological analysis. The weighting schemes consist of the Boolean model that uses 0 to refer to the absence of the word or 1 to its presence, word count, normalized word count, term pruning, TF, and TF-IDF. The Arabic morphological analysis contains two stemming techniques; stemming and light stemming. Stemming substitutes words with their stems, while light stemming eliminates common affixes from words without substituting them with their stems, which preserve the meanings. In (Saad, 2010) study,

seven Arabic datasets; BBC Arabic website bbcarabic.com corpus, CNN Arabic website cnnarabic.com corpus, Open-Source Arabic Corpus (OSAC), Contemporary Corpus of Arabic (CCA), Aljazeera News corpus, Khaleej newspaper of the year 2004 corpus, and a corpus from multiple online Arabic newspapers; were used to evaluate the effects of preprocessing in depth using BOW. Results highlight the significance of preprocessing Arabic text as a method for feature reduction, which reduces complexity for classifiers, consumes less storage, and saves processing time. Findings also report the following: light stemming with term pruning is the best feature reduction technique that shows the highest performance score. SVM and variations of the NB perform better than the other algorithms, and weighting schemes influence the performance of the distance-based classifier.

In another study (Duwairi & El-Orfali, 2014), as part of building a sentiment analysis system for Arabic text, the authors analyze the effects of removing stop words and stemming on the performance of the classifier. They use two datasets; the first one contains reviews that the authors collect from political articles at Aljazeera Arabic news website, and the second one is publicly available and includes movie reviews written in MSA. The system has several schemes for BOW to represent the features and uses SVM, NB, and KNN classifiers. The results show that all pre-processing techniques improve performance (vs. no pre-processing) on the Aljazeera reviews dataset. However, for the movie reviews dataset, stemming shows a reduction in accuracy score. Authors attribute the adverse effects of stemming on performance for the movie reviews dataset to the Rapidminer's stemmer, which has high error rates. The authors also point out the fact that

the movie reviews dataset is larger than the Aljazeera reviews dataset; thus, the high error rate of the stemmer is more obvious and shows sharper effects on the movie reviews dataset.

English text has been investigated more intensively than Arabic text to study the effects of pre-processing techniques on text classification systems. In this section, I also discuss two studies of the most recent ones that focus on analyzing the effects of pre-processing techniques for English text classification tasks. HaCohen-Kerner, Miller, and Yigal (2020) study six preprocessing techniques: spelling correction, HTML tag removal, converting uppercase letters into lowercase letters, punctuation mark removal, reduction of repeated characters, and stop word removal. The experiments were performed on four datasets for text classification tasks; the four Universities Data Set (WebKB), the multi-labeled Reuters-21578 dataset (R8), the SMS Spam Collection, and the Sentiment Labelled Sentences; using BOW and three classifiers: a Bayes Networks (BN), a variant of SVM (SMO), and a Random Forest. The results demonstrate the importance of experimenting with various pre-processing techniques in systematic combinations when developing a text classification system and highlight the fact that there is no golden set of preprocessing techniques that can always improve the performance of the classifiers regardless of the dataset used.

Woo, Kim, and Lee (2020) also analyze the effect of pre-processing techniques for the text classification system. Their study applies two main types of pre-processing techniques. Typical pre-processing, includes special character elimination, lemmatization, lowering upper case, and punctuation splitting or merging. Advanced

preprocessing, includes technical terminology pre-processing and sorting sentences based on their complexity. The technical terminology pre-processing uses a customized rule-based algorithm that utilizes a technical term corpus to extract rules. These rules are used to extract technical terminology, and Wikipedia API is used to verify the accuracy of the identified terminologies. When a technical terminology is identified, the terminology is reformed to ensure proper segmentation during pre-processing and to reduce any semantic change. Sorting sentences based on complexity calculates the entropy of each sentence based on the distribution of syllables. The entropy value is used as the sentence complexity score, and used to sort sentences based on their complexity to investigate their effect on accuracy. The Stanford Natural Language Inference (SNLI) corpus was used for training and testing the model. In this study, the authors consider modern neural network models only; a CNN model, Siamese LSTM model, and a transformer model. The experiments use an evaluation method that depends on combinations of pre-processing techniques. Results report higher accuracy scores when two pre-processing techniques are applied, specifically, lemmatization and punctuation splitting, lemmatization and lowering upper case letters, or lowering upper case letters and punctuation splitting. If only one technique is applied, the authors recommend using lemmatization, lowering upper case letters, or punctuation splitting. The results also demonstrate that pre-processing techniques, which shorten the lengths of sentences, are not recommended. Special character elimination, normalization techniques, and ordering sentences based on their complexities do not improve the accuracy.

Methodology

Generally, I follow the traditional text classification pipeline in the experimental study, staring from pre-processing, feature extraction, classification, and ending with performance evaluation. During pre-processing, I apply each pre-processing technique separately; then, I create a version of the dataset that has been proceeded by all the pre-processing techniques, in addition to the available raw version without pre-processing. I apply only one form of pre-processing at a time and develop multiple classifiers.

Datasets

Two out of nine corpora presented in Chapter 6 support the experiments in this chapter. The corpora all contain tweets, as a way of controlling any effects of platform on the transfer of models. One dataset is dialectal and the other one is multi-dialect to analyze the effect of pre-processing on various dialectal levels. I narrow down the selection to only two datasets as this experiment examine the performance on several models under several text pre-processing steps, which requires huge amount of processing times and power. The first dataset is the OSACT dataset, which supports two classification tasks: (a) detecting if a post is offensive or not, and (b) identifying if a post is hate speech or not. (Mubarak et al., 2020). The second dataset is the L-HSAB, which contains 5,846 tweets labeled as hate (468 tweets), abusive (1,728 tweets), or normal (3,650 tweets) (Mulki et al., 2019).

Pre-processing Techniques

The followings are the pre-processing techniques that were included in the experiments:

1. Emoji and Emoticon Conversion

Emojis and emoticons are often used to convey feelings and attitudes, which could be very valuable to offensive language detection. Pre-processing text for emojis has been shown to improve aggression detection (Orasan, 2018).

I convert emojis to an Arabic textual label that describes the content of the emoji. For this, I used Beautifulsoup4 4.8.2[22] to extract the entire set of emojis defined by Unicode.org[23], which provides textual descriptions for emojis written in English language. I translate the descriptions to Arabic using Translate 1.0.7[24] python package. The final extracted emoji list contains 1,374 emojis. Table 16 shows examples of emojis with their Arabic textual labels along with their English translations.

Table 16 Examples of emojis and their Arabic labels with English translations

Emoji	Arabic Label	English Translation
🙂	وجه مبتسم قليلا	slightly smiling face
😃	وجه مبتسم بعيون كبيرة	grinning face with big
🐵	وجه القرد	monkey face
🐒	قرد	monkey
🎂	كعكة عيد الميلاد	birthday cake

I do not consider emoticons in this study because emoticons exhibit much more variation and need to be observed on larger corpora to determine their semantics.

2. Arabic Dialect Normalization

[22] https://pypi.org/project/beautifulsoup4/
[23] https://home.unicode.org
[24] https://pypi.org/project/translate/1.0.7/

The OSACT dataset is extracted from Twitter, in which users usually use dialectal Arabic. To normalize dialectal nouns, I convert to MSA. For instance, the word cat from multiple dialects; Egyptian is "Otta/أطة", in Levantine, it is "Bisse/بسة", in Gulf, it is "Qatwa/قطوة", in Moroccan, it is "Qetta/قطة", in Yamani, it is "demah/دمة", and in Iraqi, it is "Bazzuna/بزونة"; is converted to its MSA form "قطة". The selected dialectal nouns primarily reflect the training set of the dataset.

3. Words Categorization

This step converts hyponyms to hypernyms for a selected set of nouns. From the manual inspection for a sample of tweets from the dataset, I notice that it is very common to mention names of animals among hate speech tweets. Thus, I manually created a list of the most common animal names used in different Arabic dialects, such as "كلب/dog", "خنزير/pig", "حية/snake". The list considers the dialectal variations in animal names as well as the gender and quantity variations in animal names. During this step, all animal names are substituted by the Arabic word "حيوان", which means animal.

4. Letter Normalization

Some Arabic letters can be written in various forms depending on the location within the word. I normalize three letters to convert all their forms into one form, Alif (إ،آ،أ to ا), Alif Maqsura (ئ،ي to ى), and Ta Marbouta (ة to ه). In addition, letters repeated more than two times within a word are reduced to two times only.

5. Hashtag Segmentation

Hashtags are commonly used in user-generated content to support sharing content and highlight important phrases within the posts. The text used in writing hashtags is

written in a format that connects all words. Usually, in the English language, users capitalize the first letter of each word and keep all hashtag phrase connected without space (e.g., "#hateSpeech"). However, this capitalization method cannot apply to the Arabic language. Thus, hashtags in Arabic usually connect multiple words using '_', similar to "#hate_speech". I convert hashtags into a meaningful format that can be understood by the model by removing the "#" symbol and replace "_" by a space. For example, the hashtag "#التعاون_ الهلال/#AlHillal_Altaawen" (name of a sport team) is converted to "الهلال التعاون/AlHillal Altaawen", which is easier for the system to process.

6. Miscellaneous

In addition to the previous steps, I apply some miscellaneous pre-processing techniques that are commonly performed in the field. I remove numbers, kashida, diacritics, HTML tags, more than one space, three or more repetitions of any character, and some symbols or terms (e.g., "\"", "...", "!", "?", ".", ";", ":", "\#", "@", "\$", "\&"). Moreover, I use the list of Arabic stop words defined by Alrefaie Github[25], which contains 750 stop words to filter the text further by removing stop words.

Table 17 shows an example of a tweet from the OSACT dataset before and after each preprocessing step.

Table 17 Example of tweet using each pre-processing step

Pre-processing Step	Example Before Pre-processing	Example After Pre-processing	Translation

[25] https://github.com/mohataher/arabic-stop-words

Pre-processing Step	Example Before Pre-processing	Example After Pre-processing	Translation
Emoji and Emoticon Conversion	فى حاجات مينفعش نلفت نظركوا ليها زى الاصول كده يا اتربيتوا عليها يا لأ😇	فى حاجات مينفعش نلفت نظركوا ليها زى الاصول كده يا اتربيتوا عليها يا لأ وجه مبتسم مع هالة	We do not want to bother you with things that you are not aware about like race
Dialects Normalization	لا تعتب عليه هيدا اسمه ابو صرماي	لا تعتب عليه هيدا اسمه ابو حذاء	Don't worry his name is shoe's owner
Word Categorization	ما كان ممكن تبدأ بيه بدل ما تخسر تبديل يا حمار	ما كان ممكن تبدأ بيه بدل ما تخسر تبديل يا حيوان	You need to start with him better than losing, donkey
Letters Normalization	يا أنت يا حبيبي يا أنيق	يا انت يا حببى يا اندق	Lovely, you are elegant
Hashtags Segmentation	نحبك يا تاج الرأس يا كل أيامنا أنت #يوم_الأم	نحبك يا تاج الرأس يا كل أيامنا أنت يوم الأم	We love you, you are our days mother's day
Miscellaneous	RT @USER: @USER @USERيا نموت احرار يا نمشي ماليزيا	نموت احرار نمشي ماليزيا	We either die free or we will leave to Malaysia

Features

Due to datasets' informality, as users usually write in inconsistent formats in social media, I consider using character-based features. I develop BOW features using n-grams of 2 to 5 characters. I use the same features for all models except BERT-based models, for which I use the output from the BERT encoder as the feature. To better assess the deep learning classifiers, I use AraVec word embeddings in addition to BOW features (Soliman et al., 2017).

Classification Models

Multiple classifiers of different machine learning categories were developed. The LR and the SVM were used as representative traditional machine learning classifiers. Bagging and Random Forest represent ensemble machine learning classifiers.

The ANN classifiers; RNN and LSTM; are also included in the study. I used Keras to develop ANN models using 3000 for maximum features, 0.25 for drop-out, 100 units, 1000 for batch size, and 500 epochs. Finally, the AraBERT model (Antoun et al., 2020) and the Arabic-BERT model (Safaya et al., 2020) represent BERT-based classifiers. BERT-based language models are applied with maximum length = 128, patch size = 16, epoch = 2, epsilon = 1e-8, and learning rate = 2e-5. For both BERT models, I do not use feature engineering, instead, I use the pool layer from the encoder and feed it into an FFNN layer.

Results

I consider macro measurements for all metrics; precision, recall, accuracy, and F1; because the datasets are imbalanced. Overall, results do not demonstrate any significant patterns for specific Arabic text prepossessing steps in all tasks. The variations in performance achieved by different preprocessing steps for each classifier are slightly small.

Figure 37 and Figure 38 show results for offensive language detection using the OSACT dataset. As can be seen from Figures (a) to (d), the SVM and the Random Forest perform better when applied to fully processed text, while the LR and Bagging perform better when applied to dialect normalized text. The deep learning classifiers show

different performance results when applied using BOW features versus AraVec features, as can be noticed from (e) and (f), on general preprocessing text reduces the performance of BOW-based classifiers. For Figures (g) and (h), miscellaneous preprocessing and hashtag segmentation improve the performance of AraVec-based deep learning classifiers. Both BERT-based models in (i) and (j) show better performance when applied to dialect normalized text, while AraBERT also reports improved performance when applied to emoji and emoticon converted text.

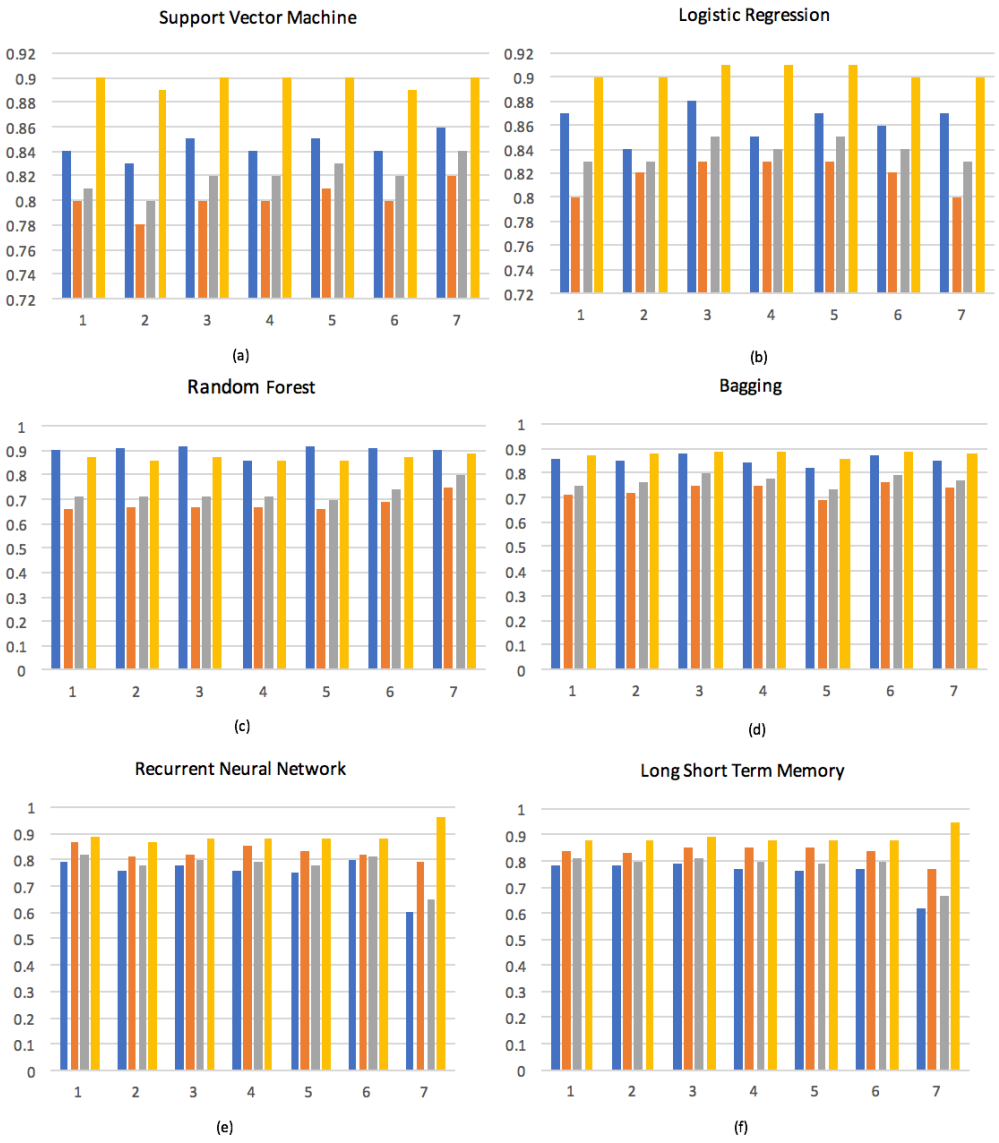
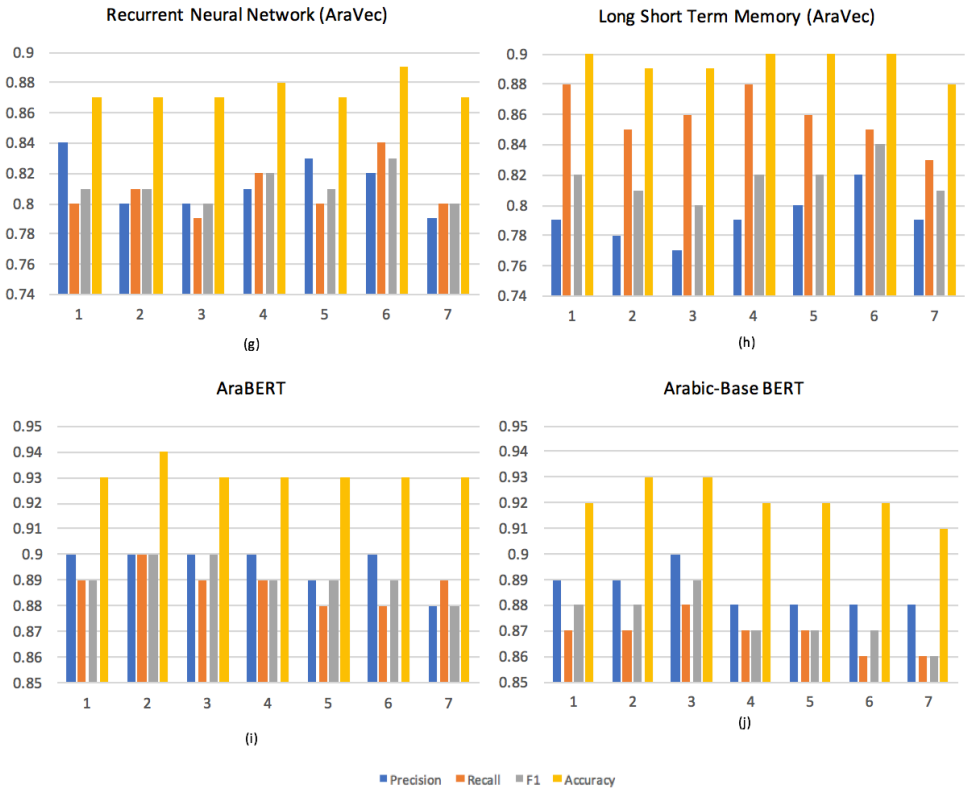


Figure 37 Results for offensive language detection using the OSACT dataset (part 1)



Step Number	Description
1	Baseline Model
2	Emoji and Emoticon Conversion
3	Dialects Normalization
4	Word Categorization
5	Letters Normalization
6	Miscellaneous and Hashtags Segmentation
7	All Preprocessing Steps

Figure 38 Results for offensive language detection using the OSACT dataset (part 2)

Results for hate speech detection using the OSACT dataset are presented in Figure 39 and Figure 40. The SVM, Figure (a), shows very high performance when applied to fully processed text; however, the LR reports the lowest performance when applied to fully processed text. As can be seen from Figure (b), the LR performs better with miscellaneous pre-processed text and hashtag segmentation. Figure (c) for the Random Forest demonstrates similar effects on performance by the unprocessed text and

the fully processed text. Bagging, Figure (d), and all deep learning classifiers from Figure (e) to (h) gain in performance when applied with miscellaneous pre-processed text and hashtag segmentation pre-processing. Bert-based models, Figures (i) and (j), do not show improvement of pre-processed text over unpreprocessed text.

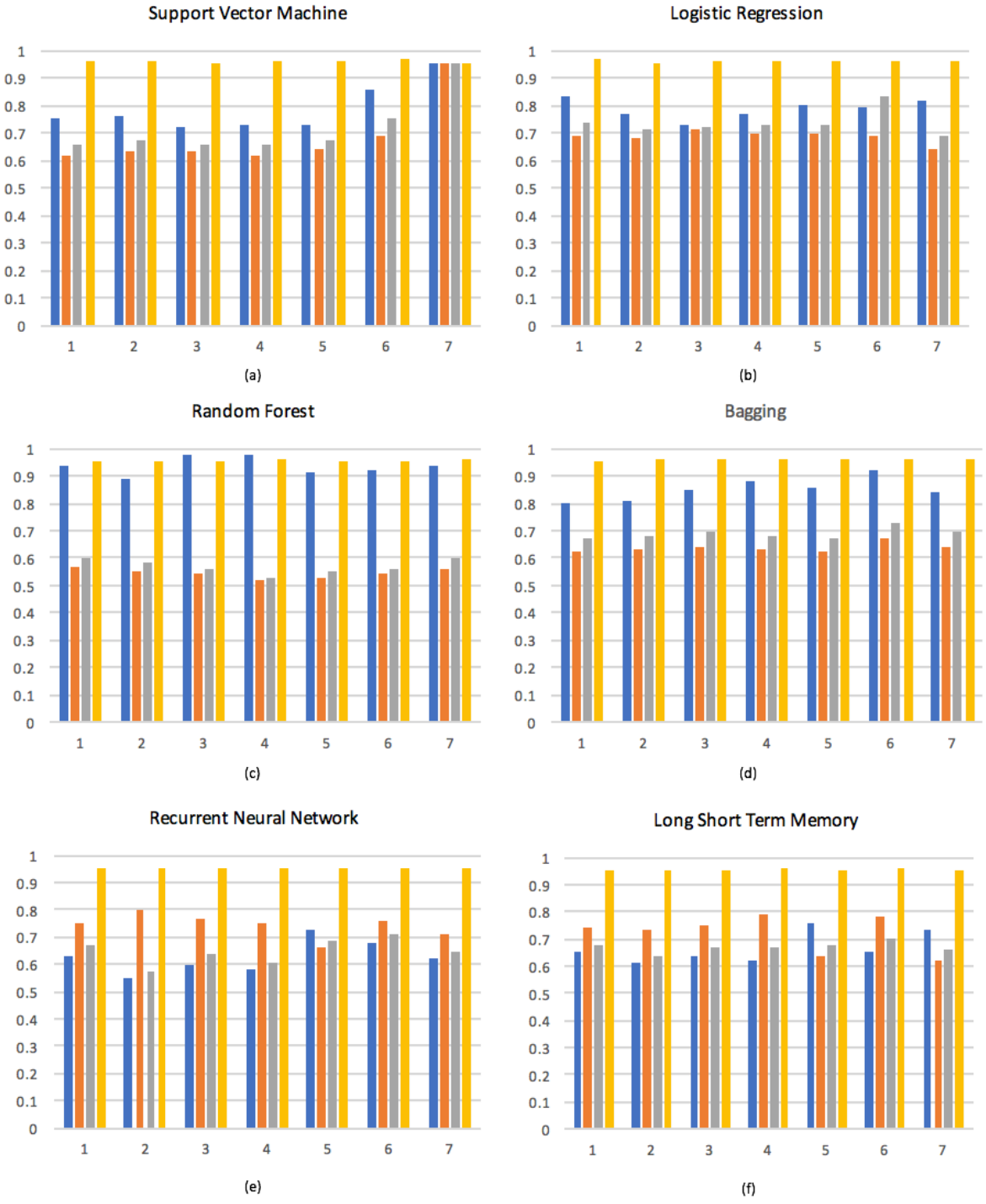
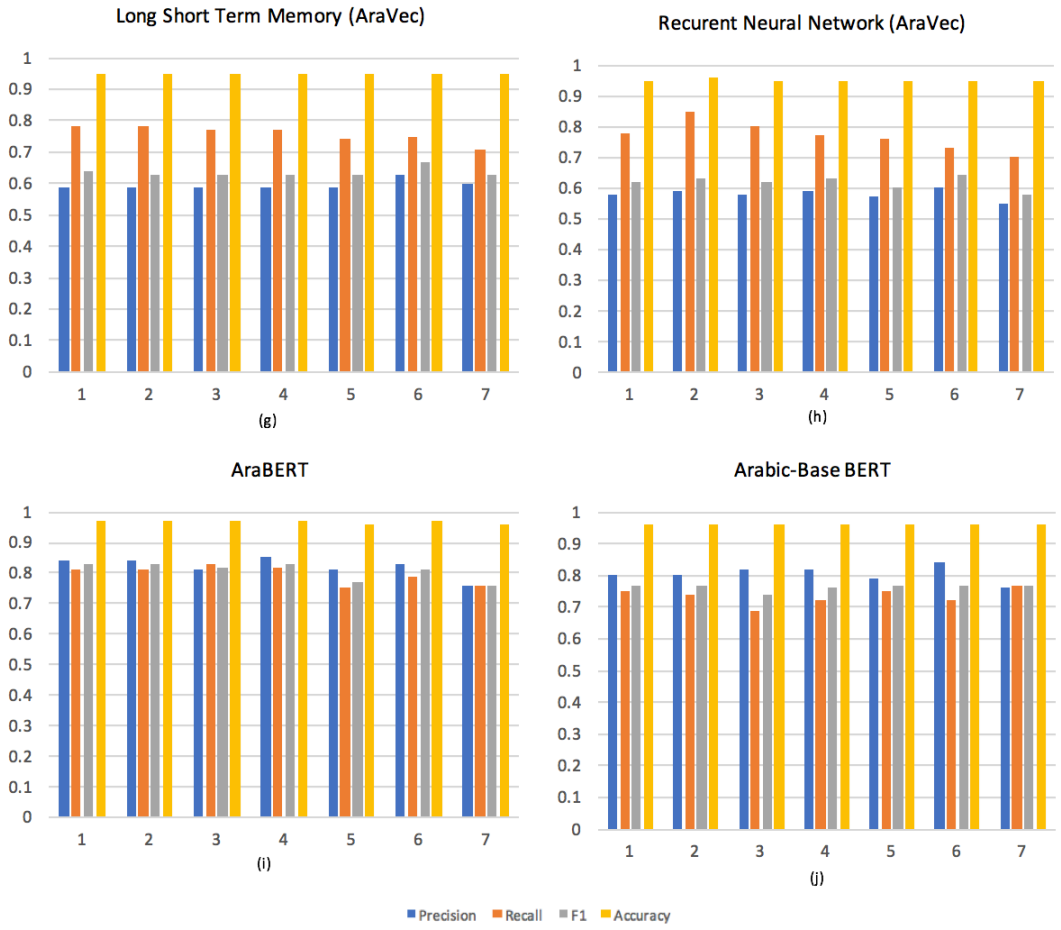


Figure 39 Results for hate speech detection using the OSACT dataset (part 1)



Step Number	Description
1	Baseline Model
2	Emoji and Emoticon Conversion
3	Dialects Normalization
4	Word Categorization
5	Letters Normalization
6	Miscellaneous and Hashtags Segmentation
7	All Preprocessing Steps

Figure 40 Results for hate speech detection using the OSACT dataset (part 2)

Graphs in Figure 41 and Figure 42 highlight the results of abusive and hate speech detection using the L-HSAB dataset. Figure (a) for the SVM demonstrates no improvement from pre-processing, while Figure (b) for the LR reports slightly better performance when applied to dialect normalized text and letter normalized text over un-

pre-processed text. The Random Forest: Figure (c), shows better performance using dialect normalized text and fully processed text. On the other hand, Bagging in Figure (d) does not highlight any improvement from pre-processing text. The RNN model with BOW, Figure (e), demonstrates no improvements in performance by any pre-processing and performs best on as unpreprocessed data. Figure (f) for the LSTM model with BOW obtains the best performance on unpreprocessed text, dialect normalized text, and miscellaneous pre-processed and hashtag segmented text. AraVec-based models report different results, in Figure (g) for the RNN, with the best performance achieved on miscellaneous pre-processed and hashtag segmented text. In Figure (h) for the LSTM, best performance is obtained on un-preprocessed text. Figure (i) for AraBERT model reports the same highest performance score by four pre-processing steps: emoji and emoticon conversion, dialect normalization, word categorization, and miscellaneous pre-processing and hashtag segmentation. Arabic-BERT model demonstrates slightly better performance after emoji and emoticon conversion and dialect normalization over un-preprocessed text.

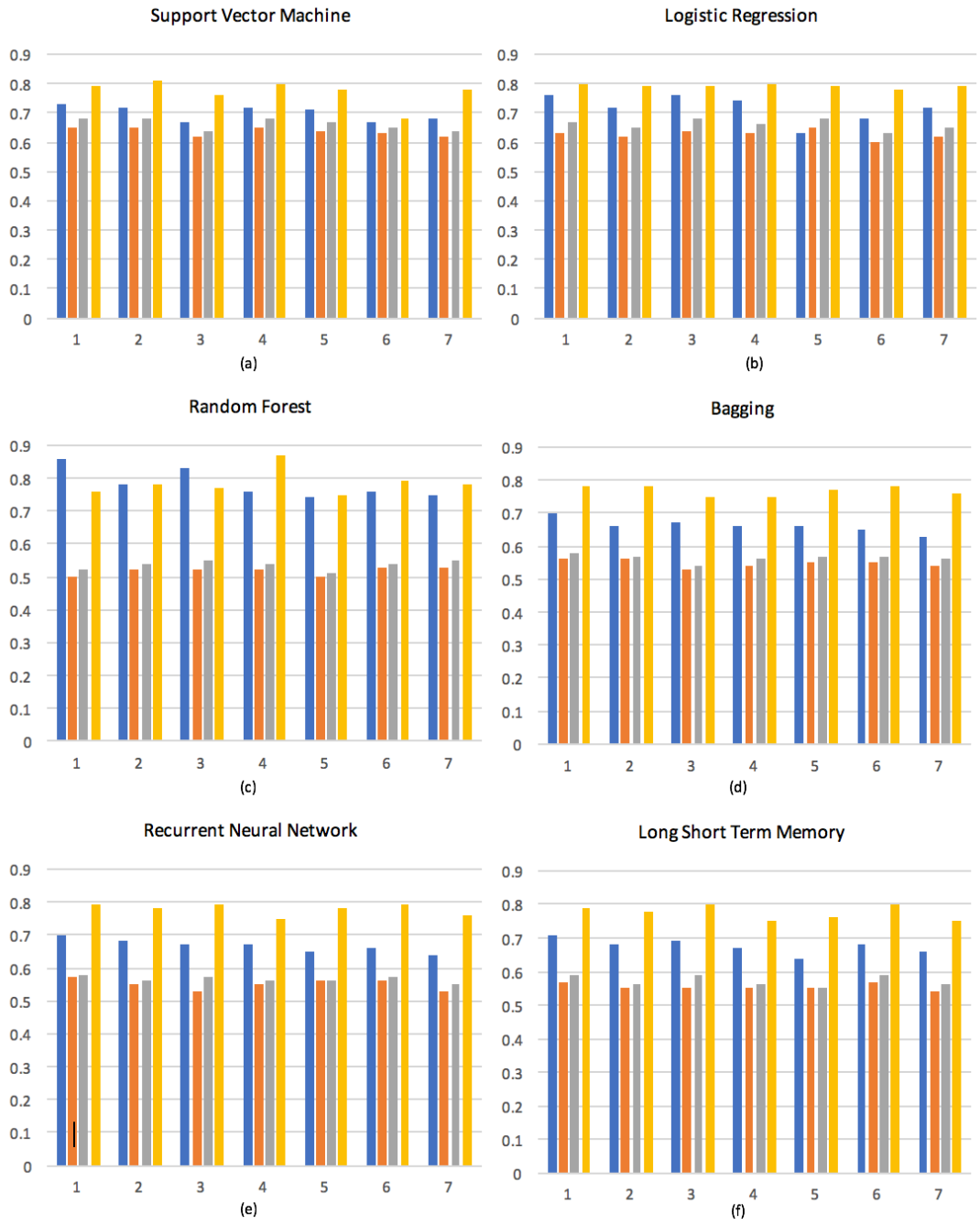
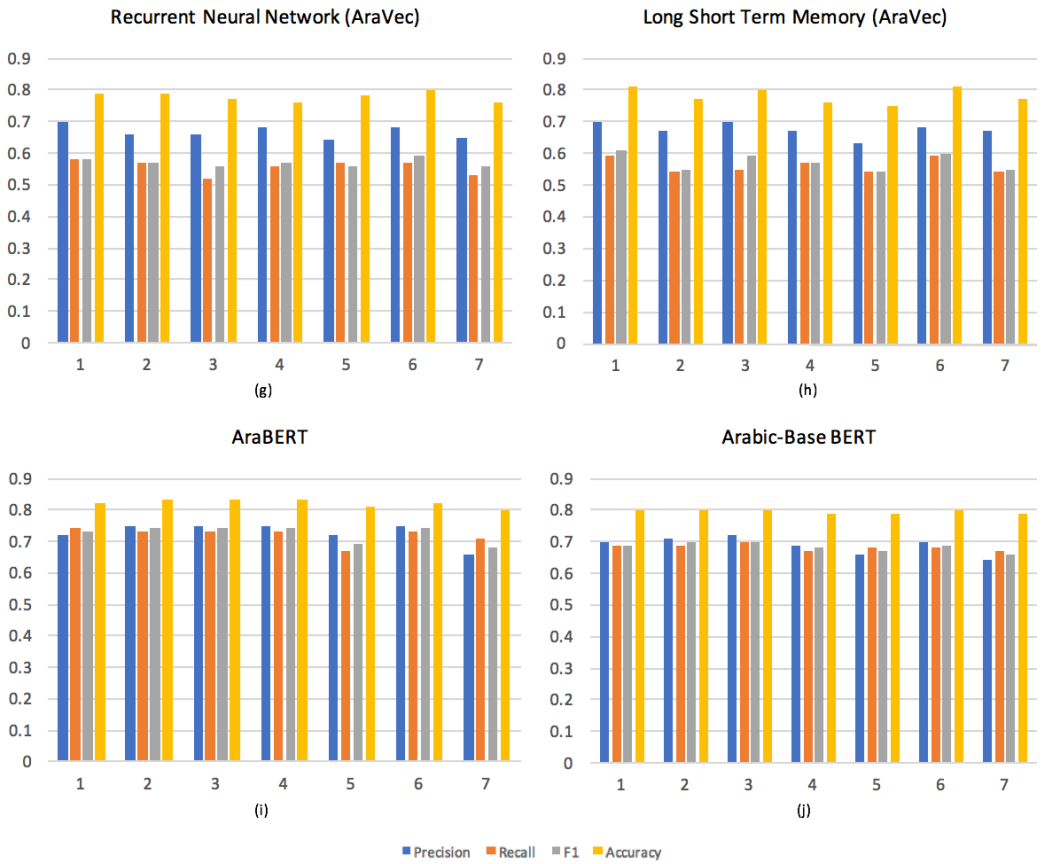


Figure 41 Results for abusive and hate speech detection using the L-HSAB dataset (part 1)



Step Number	Description
1	Baseline Model
2	Emoji and Emoticon Conversion
3	Dialects Normalization
4	Word Categorization
5	Letters Normalization
6	Miscellaneous and Hashtags Segmentation
7	All Preprocessing Steps

Figure 42 Results for abusive and hate speech detection using the L-HSAB dataset (part 2)

Statistical Analysis of Results

I perform statistical analysis for the macro-F-1 score results to summarize results from the three tasks; offensive language detection for the OSACT dataset, hate speech detection for the OSACT dataset, and abusive and hate speech detection for the L-HSAB dataset. The statistical analysis includes minimum, maximum, mean, standard deviation,

and range of macro-F1 scores for the seven preprocessing steps presented on the previous section. Table 18, Table 19, and Table 20 show the statistical analysis for the results from the three tasks.

Table 18 Statistical analysis of macro-F1 results for offensive language detection using the OSACT dataset

Classifier	Minimum	Maximum	Mean	Standard Deviation	Range
SVM	00.8	0.84	0.82	0.01	0.04
LR	0.83	0.85	0.84	0.00	0.02
Random Forest	0.70	0.80	0.72	0.03	0.10
Bagging	0.73	0.80	0.77	0.02	0.07
RNN (BOW)	0.65	0.82	0.77	0.06	0.17
LSTM (BOW)	0.67	0.81	0.78	0.05	0.14
RNN (AraVec)	0.80	0.83	0.81	0.01	0.03
LSTM (AraVec)	0.80	0.84	0.82	0.01	0.04
AraBERT	0.88	0.90	0.89	0.00	0.02
Arabic-BERT	0.86	0.89	0.87	0.00	0.03

Table 19 Statistical analysis of macro-F1 results for hate speech detection using the OSACT dataset

Classifier	Minimum	Maximum	Mean	Standard Deviation	Range
SVM	0.66	0.95	0.72	0.11	0.29
LR	0.69	0.83	0.73	0.04	0.14
Random Forest	0.53	0.60	0.69	0.02	0.06
Bagging	0.67	0.73	0.77	0.02	0.07
RNN (BOW)	0.57	0.71	0.65	0.05	0.14
LSTM (BOW)	0.64	0.70	0.67	0.02	0.06
RNN (AraVec)	0.58	0.64	0.62	0.02	0.06
LSTM (AraVec)	0.63	0.67	0.64	0.01	0.04
AraBERT	0.76	0.83	0.80	0.03	0.07
Arabic-BERT	0.74	0.77	0.76	0.01	0.03

Table 20 Statistical analysis of macro-F1 results for abusive and hate speech detection using the L-HSAB dataset

Classifier	Minimum	Maximum	Mean	Standard Deviation	Range
SVM	0.64	0.68	0.66	0.02	0.04
LR	0.63	0.68	0.66	0.02	0.05
Random Forest	0.51	0.55	0.53	0.01	0.04
Bagging	0.54	0.58	0.56	0.01	0.04
RNN (BOW)	0.55	0.58	0.56	0.01	0.03
LSTM (BOW)	0.55	0.59	0.57	0.02	0.04
RNN (AraVec)	0.56	0.59	0.57	0.01	0.03
LSTM (AraVec)	0.54	0.61	0.57	0.03	0.07
AraBERT	0.68	0.74	0.72	0.03	0.06
Arabic-BERT	0.66	0.70	0.68	0.01	0.04

Generally, the result demonstrates the limited effects of Arabic text pre-processing on offensive language detection. In general, the traditional machine learning classifiers gain more from pre-processing than ensemble machine learning classifiers and deep learning classifiers. Only for hate speech detection from the OSACT dataset, traditional machine learning, particularly the SVM, performs better than all other classifiers, including BERT-based classifiers.

The advanced BERT-based models gain the least from pre-processing. Therefore, these models can be applied on raw text without applying any pre-processing. Examining the values of standard deviation from the statistical analysis tables shows that the

majority of experiments report a very small value of 0.01, which indicates that macro-F1 scores of the seven pre-processing steps are clustered very closely around the mean. Among all tables, the largest standard deviation is 0.11, which is the result of the SVM classifier for hate speech detection.

The statistical analysis tables also report the lowest macro-F1 scores for all tasks by both ensemble machine learning classifiers, Random Forest and Bagging. However, the classifier with the best performance varies from task to task. For instance, AraBERT performs best on both offensive language detection from the OSACT dataset and abusive and hate speech detection from the L-HSAB dataset, while the SVM performs best on hate speech detection from the OSACT dataset.

For offensive language detection from the OSACT dataset, the highest macro-F1 score is 0.90 by the AraBERT classifier. Table 18 shows that the highest variations among the seven preprocessing steps occur when applying the RNN and the LSTM with BOW features for offensive language detection. Thus, further considerations to the pre-processing phase for deep learning systems with the BOW feature is important for offensive language detection in Arabic. Table 19 presents the statistical analysis results for hate speech detection from the OSACT dataset, it highlights the effectiveness of applying the SVM classifier. Moreover, the variation is relatively high among the seven pre-processing steps for the SVM classifier in comparison to the rest of the classifiers, which indicates the significance of Arabic text pre-processing procedures to the SVM classifier for this task. In Table 20, the variations become more even, most of the

classifiers have 0.01 standard deviation, and BERT-based classifiers significantly exceed the others in performance.

Chapter Summary

In this chapter, I examine the impact of seven pre-processing steps on Arabic offensive language detection. I focus on seven pre-processing steps: conversion of emoji to texts, hashtag segmentation, normalization of different forms of Arabic letters, normalization of selected nouns from Arabic dialects to MSA, conversion of selected hyponyms to hypernyms, and basic cleaning processes such as removing numbers, kashida, diacritics, and HTML tags. The experiments cover several classifiers, including traditional machine learning classifiers, ensemble machine learning classifiers, deep learning classifiers with BOW and AraVec, and BERT-based classifiers. The results show unequal effects of pre-processing steps on the classifiers and demonstrate the limited gains from Arabic text pre-processing on offensive language detection. Thus, customization of the pre-processing procedures is recommended based on the results. The advanced BERT-based classifiers show very limited gains from pre-processing while performing well on offensive language detection, which advocate for skipping pre-processing altogether. We therefore utilize BERT-based models without pre-processing in the following chapters.

TRANSFER LEARNING ACROSS-DIALECTS

" وَلَا تَسُبُّوا الَّذِينَ يَدْعُونَ مِن دُونِ اللَّهِ"
“And do not insult those they invoke other than Allah (God)”

- Holy Qur’an, chapter Al-An’am (6), verse 108.

The previous chapter highlights the advantage of using AraBERT for Arabic offensive language detection compared to other BERT-based models. This chapter builds on the previous one and utilizes AraBERT in a way that allows Arabic dialects to inform each other in offensive language detection. The experiments build on the principles of transfer learning, introduced in the previous chapter.

More specifically, this chapter investigates the effect of continuing training and fine-tuning the AraBERT model on one set of Arabic dialects and testing it on heldout Arabic dialects. The experimental study examines the second hypothesis of this dissertation: *“H2. Creating a dialectal representation can boosts the performance of automatic offensive language detection.”*

The discussion section of this chapter answers the following questions:

- Could AraBERT transfer knowledge from one Arabic dialect to another, and boost the system performance for offensive language detection?

- Which Arabic dialect(s) contain(s) the most valuable linguistic dialectal knowledge in terms of its impact when deployed on other dialects?

Transfer Learning Cross-Dialect

Cross-dialect transfer learning is a way of estimating the generalizability of the models in terms of performance over several Arabic dialects. Thus, a selected set of dialectal datasets are used for continuing training, fine-tuning, and evaluating the model to see how one dialect informs offensive language detection on other dialects.

Methodology

In order to assess the information various dialects have about each other for offensive language detection, I build on the pre-trained AraBERT model with the parameters tuned in Chapter 6. I continue training the pre-trained model on the training set of one dialect, fine-tune to the training data of another dialect, and apply to the test data of the fine-tuned dialect. I do this for all pairs of dialects. I also check the ability of multi-dialect datasets to recognize offensive language in individual dialects. I test the significance of performance differences using the Paired Bootstrap test provided by Dror et al. (2018) and described in Chapter 3 at p-value of 0.05.

Datasets

Four out of nine corpora presented in Chapters 5 and 6 support the experiments in this chapter. The corpora all contain tweets, as a way of controlling any effects of platform on the transfer of models. One tweet corpus that is linguistically very different from the other tweet datasets is also omitted (Religious Hate Speech Dataset). The datasets that I utilize are: L-HSAB which contains Levantine tweets, Egyptian Tweets

dataset which contains Egyptian tweets, T-HSAB which contains Tunisian, and OSACT which contains tweets from a mix of dialects. I utilize the datasets which contain individual dialects to see how individual dialects inform each other. I utilize the multi-dialect dataset to see if a wealth of dialects can support individual, resource-constrained dialects. Only binary classes are applied; offensive or not offensive. Thus, I convert different types of offensive language to offensive class. For example, the L-HSAB and T-HSAB datasets differentiate between hate and abusive language classes; which were both converted to offensive class. I randomly apply 80%-20% split to all datasets to create the train-test portions. Table 21 previews the distribution of each dataset. As can be seen, the Egyptian Tweets dataset is significantly smaller than the others.

Table 21 Datasets Distribution

Dataset (Dialect)	Train	Test
Egyptian Tweets (Egyptian)	880 tweets (not offensive = 353, offensive = 527)	220 tweets (not offensive = 100, offensive = 120)
T-HSAB (Tunisian)	4,819 samples (not offensive = 3,068, offensive = 1,751)	1,205 samples (not offensive = 752, offensive = 453)
L-HSAB (Levantine)	4,676 tweets (not offensive = 2,919, offensive = 1,757)	1,170 tweets (not offensive = 731, offensive = 439)
OSACT (Multi-dialect)	8,000 tweets (not offensive = 6,403, offensive = 1,597)	2,000 tweets (not offensive = 1,606, offensive = 394)

Pre-processing

Based on findings from the previous chapter, which demonstrate that pre-processing does not improve the performance of the BERT models, all datasets were used without pre-processing.

The AraBERT Models

The experiments utilize the AraBERT (first version of AraBERT, also called AraBERTv1-base and bert-base-arabert) model from the HuggingFace library. In all classifiers, I apply the same experimental settings: maximum length = 128, patch size = 16, epoch = 2, epsilon = 1e-8, and learning rate = 2e-5. I use the pooled output from the encoder with a simple FFNN layer to build the classifiers. The experiment is developed in Python using the PyTorch-Transformers library, and evaluation metrics were developed using the Scikit-Learn Python library. Google Colab Pro was used to conduct all experiments.

Continue Pre-training

To continue pre-training AraBERT, the BertForMaskedLM class from Huggingface library is used. First, I create a LineByLineTextDataset from Huggingface library interface for reading and tokenizing the continue pre-training corpus with block size of 128 words per line and AraBERT wordPiece tokenizer. Second, I use the DataCollatorForLanguageModeling class to create batches out of the corpus and specify 15% MLM probability for training. Third, I apply the TrainingArguments class to define my hyperparameters as the following: 5e-05 learning_rate, 1 epoch, and 32 batch size.

Fourth, I call the trainer class to train the model. All hyperparameters were selected based on the recommended values by Huggingface tutorial[26].

Results and Findings

I use macro-averaged precision, recall, F1, and accuracy score to evaluate the classifiers' performance. Table 22 provides baseline performance of the original AraBERT model without further training, applied off the shelf to the three dialects. It reports the results for each dialect using its training set to fine-tune the model and its testing set to evaluate the model.

Table 22 Performance results from individual dialectal models

Fine-tuning and Evaluating Corpus (Dialect)	Precision	Recall	F1	Accuracy
Egyptian Tweets (Egyptian)	0.68	0.66	0.66	0.68
T-HSAB (Tunisian)	0.79	0.77	0.78	0.80
L-HSAB (Levantine)	**0.86**	**0.86**	**0.86**	**0.87**

Table 23 shows the performances of dialectal AraBERT models, each built by continuing training the AraBERT models on the training portion of the training dialect, fine-tuned to the training portion of the heldout dialect, and applied to the testing portion of the heldout dialect.

[26] https://huggingface.co/blog/how-to-train

Table 23 Performance results from dialectal continued pre-trained models

Training Dialect	Fine-Tuning and Evaluating Dialect	Precision	Recall	F1	Accuracy	Significant Test *
Egyptian Tweets (Egyptian)	T-HSAB (Tunisian)	0.79	0.77	0.78	0.80	0.742
	L-HSAB (Levantine)	0.83	0.83	0.83	0.84	0.670
T-HSAB (Tunisian)	Egyptian Tweets (Egyptian)	0.68	0.64	0.64	0.66	0.000
	L-HSAB (Levantine)	0.82	0.83	0.83	0.84	0.915
L-HSAB (Levantine)	Egyptian Tweets (Egyptian)	0.76	0.74	0.74	0.75	0.040
	T-HSAB (Tunisian)	0.77	0.77	0.77	0.79	0.999
Egyptian Tweets (Egyptian) and T-HSAB (Tunisian)	L-HSAB (Levantine)	0.85	0.85	0.85	0.86	0.693
Egyptian Tweets (Egyptian) and L-HSAB (Levantine)	T-HSAB (Tunisian)	0.79	0.78	0.79	0.80	0.936
T-HSAB (Tunisian) and L-HSAB (Levantine)	Egyptian Tweets (Egyptian)	0.72	0.70	0.70	0.72	0.000
Concatenation of all three dialectal datasets	Egyptian Tweets (Egyptian)	0.69	0.68	0.68	0.69	0.000
	T-HSAB (Tunisian)	0.80	0.77	0.78	0.80	0.999
	L-HSAB (Levantine)	0.84	0.84	0.84	0.85	0.585

* Significant test score is calculated by applying the Paired Bootstrap test at p-value of 0.05 between the Table 22 results and results in this table based on each dataset.

In general, Egyptian and Tunisian gain from continuing pre-training AraBERT on other dialects. Levantine consistently benefits the other dialects but does not gain in performance by training on other dialects. Significant test results demonstrate significant improvement over Table 22 results for most models except when the Egyptian Tweets are evaluated. This finding is in line with the findings from Dror et al. (2018), which highlight the reduction in test's effectivity for small test sets because of the test's

assumption that the test set distribution does not differ significantly from the population distribution. Overall, results highlight the positive impact of transferring knowledge across Arabic dialects.

The table below shows the results of continued pre-training AraBERT using the OSACT dataset to evaluate whether the use of multi-dialects can support individual, resource-constrained dialects.

Table 24 Performance results from multi-dialect continued pre-trained models

Training Dialect	Fine-Tuning and Evaluating Dialect	Precision	Recall	F1	Accuracy	Significant Test *
OSACT (Multi-dialect)	Egyptian Tweets (Egyptian)	0.56	0.51	0.36	0.47	0.000
	T-HSAB (Tunisian)	0.66	0.57	0.54	0.66	0.000
	L-HSAB (Levantine)	0.73	0.56	0.50	0.66	0.000
OSACT (Multi-dialect) and all three dialectal datasets	Egyptian Tweets (Egyptian)	0.78	0.78	0.78	0.78	1.000
	T-HSAB (Tunisian)	0.79	0.80	0.80	0.81	1.000
	L-HSAB (Levantine)	0.86	0.87	0.87	0.87	0.990

* Significant test score is calculated by applying the Paired Bootstrap test at p-value of 0.05 between the Table 22 results and results in this table based on each dataset.

As can be noticed from Table 24, the results report significant improvement for all three dialects' evaluation performance after adding the training sets from each dialectal dataset to the multi-dialect corpus to continue pre-train AraBERT.

Error Analysis

Table 25, Table 26, and Table 27 show samples of tweets from each dataset with their actual and predicted labels by each model.

Table 25 Sample tweets with their actual and predicted labels by models for the Egyptian Tweets dataset (X: wrong prediction, √ correct prediction)

Tweet	Actual Label	Baseline AraBERT	Levantine Model	Tunisian Model	Levantine and Tunisian Model	Concatenated Model	Multi-Dialect and Three Dialects
التهنئة تكون للمسلمين اللذين لايعادون الاسلام ويشوهون صورته وأهله؛ أما أنت فقد بعت دينك بغرض من الدنيا؛إلى جهنم وبئس المصير *Congratulations are given to Muslims who do not hate Islam and distort its image and its people. As for you, you sold your religion for the purpose of this world, go to hell and misery of fate*	Offensive	X	√	√	√	√	√
"اهدى بس عشان أول لما ها يدعوا لتعديل الدستور أنت أول واحد هاتقول نعم" *"Be quiet, first of all, when they call for amending the constitution, you will be the first one who will say yes."*	Offensive	√	X	√	√	√	√
...محمد البرادعي رمز افتخر به ولكن الاغلب يجهلون مواقف هذا الرجل العظيم. *Mohamed El-Baradei is a symbol I am proud of, but most are ignorant of the positions of this great man.*	Not offensive	√	√	X	√	√	X
ما هو لو كام موجود علي	Not	X	√	√	X	√	√

الساحه المصريه دلوقت كنت مش هصدقه *Had he been on the Egyptian scene, I would have not believed him*	offensive						
ومتى تخلعون الكذب والنفاق عن وجوهكم *When you put off lying and hypocrisy from your faces*	Offensive	√	√	√	√	X	X
الله ينصرك ربي ويعزك ويرفعك انت وكل مخلص قابع بالمعتقلات ظُلماً وزورا *May God grant you victory, may my Lord comfort you and exalts you and every sincere person lying in the detention camps unjustly and unfairly*	Not offensive	X	X	X	X	X	X
"انزل يا حمدين واوعدك حتكون شهيد وتقابل الشهداء وحياه امي لو شفتك لعملها" *"Come down, Hamdeen, and I promise you that you will be a martyr and meet the martyrs. I swear by my mother, I will do that."*	Offensive	√	√	√	√	√	√

Table 26 Sample tweets with their actual and predicted labels by models for the L-HSAB (Levantine) dataset (X: wrong prediction, √ correct prediction)

Tweet	Actual Label	Baseline AraBERT	Egyptian Model	Tunisian Model	Egyptian and Tunisian Model	Concatenated Model	Multi-Dialect and Three Dialects
ما حدا باعصك الا اللواء No one other than the Major General is hurting you	Offensive	X	√	√	√	√	√
خازوق فيك ولي بشد عمشدك عم بيحكي عن جعجع شو خص سعد الحريري ولا عملكن	Offensive	√	X	√	√	√	√

عقدة *Screw you and people who think like you. He was talking about Geagea and not Saad Hariri, but you guys want any excuse to blame Hariri.*							
لنك رخيص *Because you are worthless*	Offensive	√	√	X	√	√	√
قل كلمتك وامشي ولايهمك كلام الحاقدين والمتزلفين كن مع الحق ولاتبالي *Say your word and walk, and do not care about haters' talk. Be with the truth and do not care*	Not offensive	√	√	√	X	√	√
هلق كله ضرب عالفلسطينيين فليش التدقيق *All hit are on the Palestinians, why caring*	Not offensive	√	√	√	√	X	X
كتلة الاخوة كتّاب العدل بيطلعلهم وزير بالحكومة؟ عقدة بيت نعيم *Will the notaries in the judiciary community get to have a representative in the government? This is the dilemma of the Naeim family.*	Offensive	X	X	X	X	X	X
انضب يا حقير منعرفكن كلاب مين يلعنك واطي *Go home you low life. We know the kind of people you adhere to, damn you.*	Offensive	√	√	√	√	√	√

Table 27 Sample tweets with their actual and predicted labels by models for the T-HSAB (Tunisian) dataset (X: wrong prediction, √ correct prediction)

Tweet	Actual Label	Baseline AraBERT	Levantine Model	Egyptian Model	Levantine and Egyptian Model	Concatenated Model	Multi-Dialect and Three Dialects
تعمل وشعب تونسي بهيم هههههه *Working and Tunisian people are bad Hahaha*	Offensive	X	√	√	√	√	√
خرجتو زين العبدين وجبتو عميل الماسون *The Masonic agent went out and got Zain Al-Abideen*	Not offensive	√	X	√	√	√	√
يعيشكم خاوتي توانسة ممكن تشرفوني قناتي ومردودين *Long live my Tunisian brothers, can you honor my channel and return*	Not offensive	√	√	X	√	√	√
لوكان جات بلحق نضرة الإسلام ماكنتيش تحكي ياغبيا عفوا مقصودي بيا *If he had come with the freshness of Islam, you wouldn't say, "I am sorry," excuse me*	Not offensive	√	√	√	X	√	√
اكبر كذابة عرفتها البشرية حاسبة روحها حاجة رلكي متسوايش بصلة يل منافقة جسوسة *The biggest liar known to mankind, You are nothing*	Offensive	√	√	√	√	X	X
برافو بشير حق المرة فاتت هوما فتو المرأة يداويها راجل عطونا فتوة جديدة تنو صوت المرأة عورة ربي يهديهم *Bravo Bashir, the truth of the time. The woman's fatwa, was healed by a man who gave us a new fatwa*	Offensive	X	X	X	X	X	X
تكفيري قح العياذ بالله وجود للسلفية الجهادية العلمية غيرهما السلفية هدي النبي صلى وسلم خربتم ديننا بارك *Radical Takfiri, God forbid*	Offensive	√	√	√	√	√	√

To better understand the variations among each model's result, Table 27 depicts the behavior of each model and how they are similar and different in understanding the data. Each cell in the table contains samples that are misclassified by its intersecting column and row's models and correctly classified by the other models. This analysis supports identifying points of failures for each model.

Table 28 Misclassified samples across the models

Models	Egyptian Model	Tunisian Model	Levantine Model	Concatenated Model
Egyptian Model	**Levantine:** لو في دولة محترمة حالا ما كنا صوتك منسمعو ع قولة زياد الرحباني مجرص الحزب فيك *If we were living in a respectable country, people like you wouldn't have a voice. As Ziad Rahbani said: you are embarrassing the political party you adhere to.* Offensive predicted as not offensive **Tunisian:** حرية لعرا توانسا احرار متربيين تفوو *Freedom for nacked, Tunisian, free, educated people* Offensive predicted as not offensive	**Levantine:** ليدي ديما صادق الله يسعدك حبيبتي وين ما كنت وبحب قلك خليها حلقه يا جبل ما يهزك ريح تحياتي القلبية الك *Lady Dima Sadiq, may God be pleased with you, my love, where were you, and with the love of your heart.* Not offensive predicted as offensive	**Tunisian:** عنصرية كانهم ليسو أفارقة إفريقي حني تكن أسمر هكدا براك الاخر *Racist, as if they were not African Africans when you were showing brown* Offensive predicted as not offensive	**Tunisian:** يامريم نكره برا اتلهى بصغارك خير تافها *Maryam, we hate righteousness,its better to play with your children* Offensive predicted as not offensive **Levantine:** Not available
Tunisian Model	**Levantine:** خدو هالتغريده وروحوا نامو انتزعة السهرة الهارب من العدالة بيحي وزير العدل خبصها مبارح هاجم العدل والعدالة والداخلية *I'll tweet this and sleep as my evening has been ruined. An outlaw just became the minister of justice. He messed up yesterday by attacking judges and the ministry of interior.* Not offensive predicted as offensive	**Egyptian:** من انحرافات صلاح نصر الى مخابرات الشتم و السب و تربية الشراميط. *From the deviations of Salah Nasr to the intelligence of insulting and raising sluts.* Offensive predicted as not offensive **Levantine:** طبعا الكلاب معروفي مين مقصودة *Of course dogs know who is*	**Egyptian:** ابئو قابلوني *see me* Offensive predicted as not offensive	**Levantine:** *A spy with the rank of Minister* Offensive classified as not offensive **Egyptian:** علماء السوء *Bad scholars* Not offensive predicted as offensive

		meant Offensive predicted as not offensive		
Levantine Model	**Tunisian:** لطفي العبدلي يمثلني *Lotfi Al-Abdali represents me* Not offensive predicted as offensive	**Egyptian:** مصر بتراثها و تاريخها اختزلت في السيسي *Egypt with its heritage and history reduced to Sisi* Not offensive predicted as offensive	**Egyptian:** تحيا مصر #ام_الدنيا يارب مصر دايماً من نصر لنصر احفظ بلادنا يارب واملاها من خيرك *Long live Egypt # Mother of the World, Lord of Egypt, always from victory to victory.* Not offensive predicted as offensive **Tunisian:** برا شيط *go shit yourself* Offensive predicted as not offensive	**Tunisian:** المغرب الكبير الدول العربية والأعجمية يجب فتحها للأسف *The Grand Morocco, Arab and foreign countries, must unfortunately be conquered* Offensive predicted as not offensive **Egyptian:** نكبة ليه بس هو إحنا بنستمد الشعارات والطهارة غير منها وهى ورقة التوت اللى بنغطى بيها مساؤنا وأفعالنا دى نعمة لكتير مش نكبة *Why a catastrophe, but it is that we derive slogans and purity other than from it, and it is the berries leaf with which we cover our misfortunes and our actions, this is a blessing for many, not a calamity* Offensive predicted as not offensive
Concatenated Model				**Tunisian:** يسترعلى الامه الاسلاميه ربي ويطرح مجلس النواب شرك شرك شرك كبيروخروج ديننا الاسلامي *The Islamic Ummah is covered up, my Lord, and the House of Representatives proposes a major polytheism, and the exit of our Islamic religion* Not offensive predicted as offensive **Levantine:** الاسد لا حضور له مع الخراف *The lion (Al-Assad) does not attend with the sheep* Offensive predicted as not offensive **Egyptian:** للاسف الميليشيات العفنه

ضيعت القضية الفلسطينية !!
Unfortunately, the rotten militias lost the Palestinian conflict.
Offensive predicted as not offensive

Generally, as can be noticed from the tables above, misclassified samples contain several types of nouns; names of figures, animals, countries, organizations, etc. Moreover, incorrectly labelled samples from the Tunisian dataset (T-HSAB) were detected, such as "المغرب الكبير الدول العربية والأعجمية يجب فتحها للأسف / The Grand Morocco, Arab and foreign countries, must unfortunately be conquered", which is labeled as offensive and was misclassified by concatenated model and the Levantine model. Similarly, the Levantine dataset (L-HSAB) includes some incorrectly annotated samples, for instance "انت واشكالك عباد المال تبحثون عن الازمات لجل مصالح شخصيه / You and your fellows are money slaves are looking for crises for personal interests"

The Tunisian model fails in classifying simple offensive Levantine tweets, containing explicit offensive words, which were correctly classified by all other models, such as "طبعا الكلاب معروفي مين مقصودة / Of course dogs know who is meant".

The findings from the analysis demonstrate that the Egyptian model and the Tunisian model demonstrate limited performance in classifying the long Levantine samples. Similarly, the Levantine model misclassifies short Tunisian samples, containing one or two words only, such as "برا شيط / go shit yourself", while the other models were better in classifying similar samples.

Chapter Summary

The experiment conducted in this chapter evaluates the impact of transfer learning using AraBERT across three Arabic dialects; Levantine, Tunisian, and Egyptian. Results demonstrate the significant impact of transferring linguistic knowledge across dialects. Based on the findings from this experiment, the next chapter explores transfer learning for user-generated content across platforms.

TRANSFER LEARNING CROSS-PLATFORM

" وَجادِلهُم بِالَّتي هِيَ أَحسَنُ"
“And argue with them in the best of manners”

- Holy Qur’an, chapter An-Nahal (16), verse 125.

Transfer learning across Arabic dialects is discussed in the previous chapter; this chapter follows a similar approach, except that its scope covers the social media platforms for user-generated content rather than dialects. Different social media platforms support different features and goals, which might affect the demographic of their users. Therefore, the type of language and content also differ among platforms, including offensive language content.

In this chapter, I describe an approach which assumes that each platform has a specific culture and language created by the structure of its community and its supported features. Thus, dialect and linguistic features used in the content are related to the platform.

This chapter answers questions related to the following:

- Does AraBERT work better for a specific social media platform?
- Does transfer learning across-platforms impact the performance of the classifier?

Sections in this chapter start with an introduction to the social media platforms used in Arabic-speaking countries. The methodology section includes a description of the datasets used in this study and the AraBERT model. Results and discussion of the findings discuss errors and system implications. The chapter concludes with a summary.

Social Media Platforms

Social media platforms host a wealth of online user-generated content. These platforms serve different goals; for instance, Instagram is mainly for sharing pictures, Facebook is for social networking, YouTube is for sharing videos, and Twitter is for microblogging. Figure 43and Figure 44 show the variety of social media platforms used by youth Arab users across the Middle East. Figure 43 highlights the variation in daily usage of WhatsApp, Facebook, Instagram, YouTube, Snapchat, and Twitter among youth users from the Gulf Cooperation Council (GCC), north Africa, and levant. Arabic youth users from different regions have distinctive rating to different platforms in term of their importance, as can be seen in Figure 44.

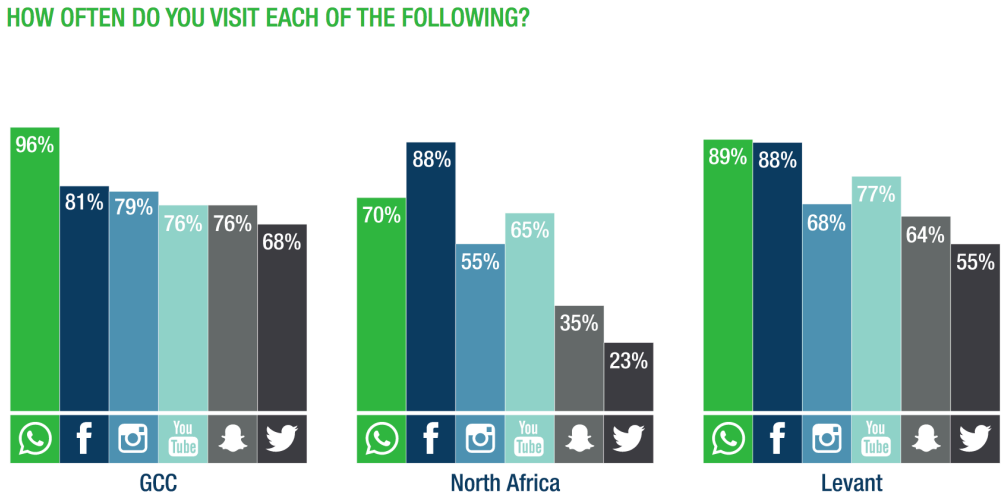


Figure 43 The percentage of youth Arabs between the age of 18-24 years' daily usage of social media during 2019 (ASDA'A BCW, 2020, p.70)

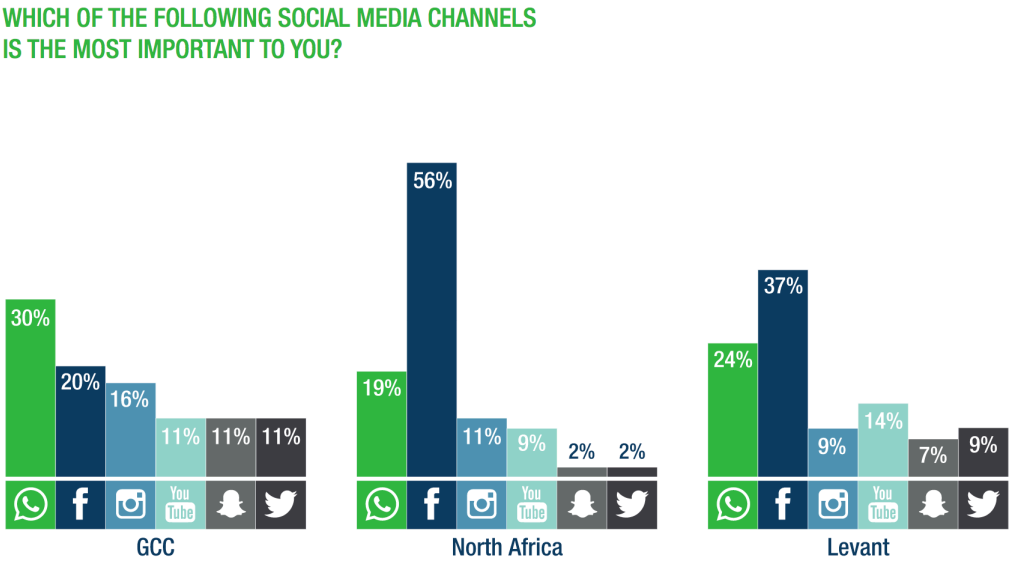


Figure 44 The percentage of youth Arabs between the age of 18-24 years' rating of the importance of social media during 2019 (ASDA'A BCW, 2020, p.70)

According to a study by Radcliffe and Abuhmaid (2020) for the users Social Media platforms in the Middle East, governments and platform owners report 44% more scrutinized of Social Media usage in 2019. The same study also reports an increase in youth Arab users among all platforms and the popularity of Facebook among Egyptians. The study highlights a general decrease in Twitter usage among Middle Eastern users except in Saudi Arabia and Turkey, which ranked the fifth and sixth largest Twitter markets.

Methodology

In order to assess the information various platforms have about each other for offensive language detection, I build on the pre-trained AraBERT model with the parameters tuned in Chapter 6. I continue training the pre-trained model on the training set of one platform, fine-tune to the training data of another platform, and apply to the test data of the fine-tuned platform. I do this for all pairs of platforms, and also groups of platforms. I also check the ability of multi-platform datasets to recognize offensive language in individual platform. I test the significance of performance differences using the Paired Bootstrap test provided by Dror et al. (2018) and described in Chapter 3 at p-value of 0.05.

Datasets

Five out of nine corpora presented in Chapter 6 support the experiments in this chapter. The corpora all contain multi-dialect samples, as a way of controlling any effects of dialect on the transfer of models. One tweet corpus that is linguistically very different from the other tweet datasets is also omitted (Religious Hate Speech Dataset). The

datasets that I utilize are: (1) Aljazeera.net Deleted Comments, which contains comments from Aljazeera News; (2) YouTube Comments, which contains comments from YouTube channels; (3) OSACT offensive and not offensive classification samples, which contains tweets; (4) MPOLD, which contains offensive samples from multiple platforms; and (5) Multi-Platform Hate Speech, which contains hate speech samples from multiple platforms. I utilize the datasets which are extracted from one platform to see how each platform inform others. I utilize the multi-platform dataset to see if a wealth of corpus from several platforms can support individual, resource-constrained platform. As before, I use only binary classes; offensive or not offensive. I convert different types of offensive languages to offensive class. For some datasets provided in train/evaluation/test split format, I merge all parts into one dataset, and then, I randomly apply 80%-20% split for train-test datasets. This supports consistency in the setting among all datasets used in this study as most of them are provided in one part. All datasets were used without any preprocessing. Table 29 provides a summary of the characteristics of each dataset.

Table 29 Datasets Distribution

Dataset	Source	Train	Test
Aljazeera Deleted Comments	Aljazeera.org News	25,353 comments (not offensive = 4,528, offensive = 20,825)	6,339 comments (not offensive = 1,125, offensive = 5,214)
YouTube Comments	YouTube	12,040 comments (not offensive = 7,379, offensive = 4,661)	3,010 comments (not offensive = 1,858, offensive = 1,152)
OSACT	Twitter	8,000 tweets (not offensive = 6,403, offensive = 1,597)	2,000 tweets (not offensive = 1,606, offensive = 394)
MPOLD	Multi-Platform	3,200 samples (not offensive = 2,660, offensive = 540)	800 samples (not offensive = 665, offensive = 135)
Multi-Platform Hate Speech	Multi-Platform	16,004 samples (not offensive = 7,973, offensive = 8,031)	4,001 samples (not offensive = 2,027, offensive = 1,974)

The AraBERT Model

I utilize the AraBERT model from the HuggingFace library. Experiment settings include hyperparameters tuned in Chapter 6: maximum length = 128, patch size = 16, epoch = 2, epsilon = 1e-8, and learning rate = 2e-5. I use the pooled output from the encoder to use with a simple FFNN layer to build the classifier. The experiment was developed in Python using the PyTorch-Transformers library, and evaluation metrics were developed using the Scikit-Learn Python library. Google Colab Pro was used to conduct all experiments.

Results and Discussion

Results

I use macro-averaged precision, recall, and F1, and accuracy score to evaluate the performance of the classifiers. The following table (Table 30) shows baseline

performance of the original AraBERT model on each platform. It reports the results on each dataset of the pre-trained AraBERT model which is fine-tuned to the training set of the target platform and applied the test set to evaluate the model.

Table 30 Performance results

Fine-tuning and Evaluating Dataset (Platform)	Precision	Recall	F1	Accuracy
OSACT (Twitter)	**0.83**	**0.82**	**0.82**	**0.86**
YouTube Comments (YouTube)	0.86	0.86	0.86	0.87
Aljazeera News Comments (Aljazeera)	0.78	0.74	0.76	0.87

As can be noticed from Table 30 above, the highest recorded macro-averaged recall, macro-averaged F1, and accuracy scores are shown for the Twitter OSACT dataset. It is also noticeable that the Aljazeera dataset has the lowest overall performance scores.

Given the baseline results of the pre-trained AraBERT model, I next adapted this model to each platform by continuing pre-training on each. To assess whether the adapted models transfer across platforms, I fine-tuned the adapted models to each of the platform's training data and applied to the corresponding test data. Table 31 records the results.

Table 31 Performance results from continued pre-trained models

Training Dataset (Platform)	Fine-Tuning and Evaluating Dataset (Platform)	Precision	Recall	F1	Accuracy	Significant Test*
OSACT (Twitter)	YouTube Comments (YouTube)	0.83	0.82	0.82	0.83	0.046
	Aljazeera News Comments (Aljazeera)	0.75	0.69	0.71	0.85	0.990
YouTube Comments (YouTube)	OSACT (Twitter)	0.80	0.72	0.75	0.96	0.000
	Aljazeera News Comments (Aljazeera)	0.76	0.70	0.72	0.86	0.930
Aljazeera News Comments (Aljazeera)	OSACT (Twitter)	0.80	0.79	0.80	0.96	0.020
	YouTube Comments (YouTube)	0.86	0.86	0.86	0.86	0.014
OSACT (Twitter) and YouTube Comments (YouTube)	Aljazeera News Comments (Aljazeera)	0.76	0.69	0.72	0.86	0.904
Aljazeera News Comments (Aljazeera) and YouTube Comments (YouTube)	OSACT (Twitter)	0.80	0.78	0.79	0.96	0.018
Aljazeera News Comments (Aljazeera) and OSACT (Twitter)	YouTube Comments (YouTube)	0.85	0.85	0.85	0.86	0.014
All Platforms	OSACT (Twitter)	0.82	0.74	0.77	0.96	0.030
	YouTube Comments (YouTube)	0.85	0.85	0.85	0.86	0.034
	Aljazeera News Comments (Aljazeera)	0.76	0.69	0.71	0.85	1.000

* Significant test score is calculated by applying the Paired Bootstrap test at p-value of 0.05 between Table 30 results and results in this table based on each dataset.

Results from the table above demonstrate the negative impact of the transfer learning across individual platforms as macro-averaged F1 scores for all datasets are less than those reported in Table 30. This finding indicates the potential of the limited effect of the platform's domain; while Aljazeera is a news platform and others used in this experiment are social media platforms, the performance results from leveraging knowledge from news to social media or vice versa show similar patterns. Most

significant test's scores demonstrate the insignificant differences in performance after applying transfer learning across platforms; however, result from Aljazeera News Comments (Aljazeera) shows the opposite, which could be related to its size. Based on Dror et al. (2018) the Paired Bootstrap test fits small dataset size and Aljazeera test set contains 6,339 comments that might violate the test's settings.

Table 32 Performance results from continued pre-trained models using multi-platforms datasets

Training Platform	Fine-Tuning and Evaluating Platform	Precision	Recall	F1	Accuracy	Significant Test*
MPOLD	YouTube Comments (YouTube)	0.85	0. 85	0. 85	0. 86	0.010
	Aljazeera News Comments (Aljazeera)	0.75	0.69	0.72	0.85	0.977
	OSACT (Twitter)	0.83	0.74	0.78	0.96	0.001
MPOLD and all individual platforms	YouTube Comments (YouTube)	**0.87**	**0.87**	**0.87**	**0.87**	**0.078**
	Aljazeera News Comments (Aljazeera)	0.76	0.71	0.73	0.86	0.845
	OSACT (Twitter)	**0.84**	**0.80**	**0.82**	**0.97**	**0.089**
Multi-Platform Hate Speech	YouTube Comments (YouTube)	0.74	0.78	0.76	0.86	0.007
	Aljazeera News Comments (Aljazeera)	0.76	0.70	0.73	0.86	0.984
	OSACT (Twitter)	0.84	0.78	0.80	0.96	0.012
Multi-Platform Hate Speech and all individual platforms	YouTube Comments (YouTube)	0.76	0.78	0.77	0.87	0.011
	Aljazeera News Comments (Aljazeera)	0.77	0.71	0.73	0.86	0.985
	OSACT (Twitter)	0.85	0.80	0.82	0.97	0.028

* Significant test score is calculated by applying the Paired Bootstrap test at p-value of 0.05 between the Table 30 results and results in this table based on each dataset.

Recorded macro-F1 scores in Table 32 demonstrate lower or equal performance by most models, except when the model is continued pre-trained using the MPOLD and all individual training sets from YouTube Comments (YouTube), Aljazeera News Comments (Aljazeera), and OSACT (Twitter), it reports an increase in macro-F1 score by 1% from those obtained in Table 30.

Analyzing significant test's scores, Table 32 reports insignificant differences over the baseline models results for all models, except the MPOLD and all individual training sets from YouTube Comments (YouTube), Aljazeera News Comments (Aljazeera), and OSACT (Twitter), which reports significant variations over Table 22 results for YouTube Comments (YouTube) and OSACT (Twitter) test sets. The same findings from previous table regarding the validity of the Paired Bootstrap test to Aljazeera News Comments (Aljazeera) applied to Table 32 too.

Error Analysis

Table 33, Table 34, and Table 35 to 6 show samples of comments/tweets from each dataset with their actual and predicted labels by each model.

Table 33 Sample tweets with their actual and predicted labels by models for the OSACT dataset (X: wrong prediction, √ correct prediction)

Tweet	Actual Label	Baseline AraBERT	YouTube Model	Aljazeera Model	YouTube and Aljazeera Model	Concatenated Model
@USER @USER يعني يا ترمب يا مغفل كيف بدك تحارب شعب اطفالة مقاومين *@USER @USER I mean, Trump you stupid, how do you plan on fighting people whose children are resistance warriors*	Offensive	X	√	√	√	√
@USER @USER @USER وانت يا لص يا متصهين لا تمثل المجتمع الكوردي *@USER @USER @USER You, thief, Zionist, you do not represent the Kurdish community*	Offensive	√	X	√	√	√
@USER @USER اذا جماعة الرئيس وجمهورو جماعة سماسرة وشتامين ومش مخليين حدا من شرن.. شو بدك العالم تسكت يا شحاطة يا عاق نفسياً وجسدياً؟ *If the president's entourage and his followers are a bunch of brokers who insult everyone without exception. How do you expect people to remain silent about this you shoe, you mentally and physically sick person?*	Offensive	√	√	X	√	√
RT @USER: #محمد_بن_سلمان_بين_شعبه<LF>انا زعلانه منك يا محمد بن سلمان يا مُز انتا مديري اجنبي واحبه ليش تخليها سعوده ، انا تحطمت نفسياً وروح… *RT @USER: #Mohammed_Ben_Salman_among_his people <LF> I am upset with you, Muhammad bin Salman, my manager is a foreigner and I love him. Why you replace him by Saudi? I am psychologically broken*	offensive	X	√	√	X	√
@USER @USER @USER باطل. باطل. باطل. السيسى الصهيونى الخاين باطل و جنودك ياسيسي يا فرعون يا طاغية باطل و دستورك باطل و نهايتك انت و جنودك الزبالة قربت انشالله *@USER @USER @USER void. void. void. The traitorous Zionist al-Sisi and his soldiers are false, O Pharaoh, you are a vain tyrant, your*	Offensive	√	√	√	√	X

constitution is null, and your end is a trash, you and your rubbish soldiers have come close to your end						
@USER اتفوه عليكم يا اولاد الكلب يا اولاد الوسخة لو كنتم رجاله كنتم تحررون بلادكم من إيران ولكن انتم اشباه الرجال فعلا انتم المخلفون من الإعراب *@USER You are children of the dog, children of the dirty women, if you were men, you would have liberated your country from Iran, you are men from the outside only.*	Offensive	X	X	X	X	X
روح روحي ونفس انفاسي ومهجة الفؤاد *Spiritual spirit, soul of my breath, and accent of heart*	Not offensive	√	√	√	√	√

Table 34 Sample comments with their actual and predicted labels by models for the YouTube dataset (X: wrong prediction, √ correct prediction)

Comment	Actual Label	Baseline AraBERT	Twitter OSACT Model	Aljazeera Model	Twitter and Aljazeera Model	Concatenated Model
منو اللي طلع لقب الملكه لها و شگتنا شگ ترى ما يلوگ لچ ابد لا شوفه و لا چرچوبه دروحي گعدي بالبيت احسن لچ و بطلي خرابيطچ ويه العالم ما ضلت بزون ما *Whoever came up with the title of the queen for her, disturb us all. It is better for you to stay at home and stop this non-sense.*	Not offensive	X	√	√	√	√
هيه صدك احلام الان.تجيني وترعبني . بل حلم.اشوفهه والله ثور هيه احلام لو البقره شنو يمكن غلطانين بل اسماء *Is her real name is Ahlam (dreams). She will terrify me if she appears in my dream, I swear I see her as a cow or a bull. It could be that her name is wrong*	Offensive	√	X	√	√	√
نضال الجعبه ملكة على الجهلاء شرواك *Nidhal Al-Juabah is a queen for the ignorant people like you*	Offensive	√	√	X	√	√
مليييقه وكرييهه *Sillyyyy and hateee*	offensive	X	√	X	X	√
وما الحياة الدنيا إلا متاع الغرور *The life of this world is nothing but the goods of vanity*	Not offensive	√	√	√	√	X

!متخلفين هههه؟ وفرنا العقل لك *Retarded Haha ?!* *We saved the mind for you*	Not offensive	X	X	X	X	X
اهيا مدحت الفنان ماجد المهندس وهو عراقي *She praised the musician Majid Al Mohandes, who is Iraqi*	Not offensive	√	√	√	√	√

Table 35 Sample tweets with their actual and predicted labels by models for the Aljazeera dataset (X: wrong prediction, √ correct prediction)

Comment	Actual Label	Baseline AraBERT	YouTube Model	Twitter OSACT Model	YouTube and Twitter OSACT Model	Concatenated Model
مع الف سلامة. ما زاد حنون في الاسلام خردلة ولا النصارى لهم شأن بحنون *Goodbye. Affectionate in Islam is not trash, and the Christians have no affection for them*	Offensive	X	√	√	√	√
نريد الاقتباسات الأصلية، إنها لمصيبة أن محتوى تلك المخطوطات محجور لدى مؤسسات ((خيرية)) لا تُعطينا إلا الترجمة التي نترجمها لنصل إلى نسخة مشوهة من الأصل *We want the original quotes, it is a misfortune that the content of these manuscripts is confiscated by ((charitable)) institutions that only give us the translation that we translate to reach a distorted copy of the original*	Not offensive	√	X	√	√	√
في ظل (ان الدين عند الله الاسلام) وبنظرة شاملة لكل الموروث الحضاري والثقافي لجميع المخلوقات سيصاغ المشروع الحضاري العظيم. *In light of (God's religion is Islam) and with a comprehensive view of all civilizational and cultural heritage for all creatures will formulate the great civilization project*	Not offensive	√	√	X	√	√
في المجتمعات العربية يجب على الحيوان الاعتماد على نفسه ان أراد البقاء *In Arab societies, animals*	Offensive	X	√	√	X	√

must rely on themselves if they want to survive						
قال تعالى (فتوكل على الله إنك على الحق المبين) سيرو على بركة الله يا اسود دولتنا *Allah says* *(So trust in God that you are on the revealed truth)* *walk on the blessing of God* *O lions of our country*	Offensive	√	√	√	√	X
اريد تردر قناة الجريزة الاخبارية *I want the channel code of Al-Jazeera news channel*	Not offensive	X	X	X	X	X
إنت شاكو بيهم؟ روح شوف شغلك.. *Do you think about them? I want to see your job ..*	Offensive	√	√	√	√	√

The tables above show the variation among models' performance for each dataset. The YouTube dataset has multiple samples with wrong annotation as can be noticed in Table 34 in the first example; منو اللي طلع لقب الملكه لها و شگتنا شگ ترى ما يلوگ لچ ابد لا شوفه و لا چرچوبه دروحي گعدي بالبيت احسن لچ و بطلي خرابيطچ ويه العالم ما ضلت بزون ما/ Whoever came up with the title of the queen for her, disturb us all. It is better for you to stay at home and stop this non-sense. The overall topic of each dataset differs; the OSACT dataset covers broad range of topics, while the YouTube dataset is dominant by musicians' related topics and the Aljazeera dataset is dominant by political topics.

Result from the original AraBERT model indicates some limitations in classifying samples with multiple words hashtags (e.g., #محمد_بن_سلمان_بين_شعبه) and Farsi letters (e.g., گ, چ); however, the others models demonstrate better performance for such samples.

Looking over samples of the misclassified comments, I calculate percentages of misclassified comments/tweets per class for each dataset based on the experiment that generates the highest performance for the same dataset in Table 36.

Table 36 presents a summary of the error analysis. The percentages are calculated per class based on the total instances of each class for each of the four testing datasets. The most common 5 tokens are presented in the table based on their frequency order.

As can be seen from Table 36, offensive and not offensive misclassified percentages vary among the datasets. Investigating top tokens among the misclassified samples shows names of countries (e.g., Saudi Arabia, Qatar, Iraq) and names of famous people (e.g., Kadim, Ahlam, Gibran), which are in some cases refer to the first name and the last name of the same person as two separate tokens. For example, 'Kadim' and 'Sahir' are the first and the last name of the same singer, and 'Gibran' and 'Basil' are the first name and the last name of the same minister. This type of terms need to be proceeded as one term rather than separate parts because the semantic meaning of its parts might not be equivalent to the semantic meaning of the term. As a result of that, some preprocessing procedures are needed to ensure proper understanding and processing of multiple tokens terms.

Table 36 Error analysis for results from the concatenated all platforms training sets pre-trained model

Test Dataset	Misclassified %		Misclassified Top Common Tokens	
	Offensive	Not Offensive	Offensive	Not Offensive
OSACT Twitter	16%	3.5%	الله/God علوقيه/ stereotype صغير/small بنت/girl حبيبي/lovely	الله/God 😂 عيون/eyes ناس/people موضوع/topic
YouTube Comment	15%	11%	كاظم/Kadim الله/God احلام/Ahlam ساهر/Sahir عراقي/Iraqi	كاظم/Kadim الله/God ناس/people مصر/Egypt ساهر/Sahir
Aljazeera Comment	8.2%	44.7%	الله/God دولة/state عراق/Iraq شعب/nation جزيرة/Jazeera	الله/God جزيرة/Jazeera سعودية/Saudi Arabia دولة/state مسلمين/Muslims

While the original AraBERT model is pre-trained mainly on news datasets as described in Chapter 6, the results from Aljazeera News Comments were the least in macro-averaged F1 score. Twitter has a limited number of characters per tweet, while Aljazeera and YouTube allow for much longer text. Moreover, some social media features such as creating profiles, following others, mentioning usernames and tagging other users is not supported by Aljazeera commenting features. Another important factor to consider is related to the dataset extraction process, the datasets were extracted using heterogeneous criteria from different authors, which could also affect their content and reduce the overlap among them. For more details on the content and size' variations among the datasets, Chapter 4 presents an exploratory datasets analysis.

A better method to investigate the effects of across-platforms transfer learning could be implemented by extracting datasets from various platforms based on one offensive language theme, such as political offensive language or religious offensive language. Sharing the same theme among datasets can support better analysis for platform-based transfer learning because we fixed the linguistic theme parameters while evaluating platform-based parameters.

System Implications

Increasing the vocabulary size of the AraBERT model through continue pre-training does not always improve the performance of the system. Thus, finding some other methods to improve the performance is required. For example, trying to adjust AraBERT's vocabulary to support offensive language classification and finding an innovative approach to customize the model further toward the target task is a critical solution to consider.

Chapter Summary

In this chapter, I applied transfer learning across several Arabic offensive language detection datasets from multiple platforms using the AraBERT model. The results report poor performance when applying transfer learning cross-platform, including YouTube, Aljazeera News, and Twitter. The variation in platforms features affect the linguistic features of their content, which limit the benefit of transferring knowledge among platforms. The overall findings from the experiment demonstrate the importance of developing a more novel comprehensive method to pre-train the AraBERT model for

the tasks of Arabic offensive language detection rather than focusing only on the platform or dialect.

DIALECTAL AND CULTURA-BASED MODEL

Imām Jafar ibn Muhammad al-Sadiq (peace and mercy be upon them) said:
"الفُحشُ والبَذاءُ والسَّلاطَةُ مِن النِّفاقِ."
"Obscene language, foulness, and impudence are all from hypocrisy"

- Bihar al-Anwar, volume 79, number 14, page 113

Multiple attempts to develop systems for detecting online Arabic offensive language are discussed in the previous chapters. An intensive preprocessing-based approach is explored in chapter 6. Then, two different approaches for transfer learning using AraBERT model were experimented. In chapter 8, an approach based cross-dialect transfer learning using three Arabic dialects is proposed to investigate the impact of transferring knowledge across different dialects. Chapter 9 defines another system pipeline based on cross-platform transfer learning. This chapter builds its pillars based on findings from previous chapters into one novel system architecture based on customizing the AraBERT model to preserve dialectal knowledge and offensive cultural knowledge within the contextual word embedding of BERT architecture.

This chapter addresses the following questions:

- Does the customized AraBERT model capture cultural and dialectal offensive knowledge that can support offensive language detection?

- How much does the customized AraBERT model contribute to different dialects and platforms?
- Where does the customized AraBERT model fail in offensive language detection?

This chapter is organized into three main sections in addition to a chapter summary at the end. Firstly, the methodology of the study conducted in this chapter is presented. The second section reports the results from applying the proposed model to nine Arabic offensive language datasets. In the third section, error analysis is conducted to examine the points of failure and patterns that confuse the models. A chapter summary to compile findings from experiments conducted in this chapter is presented at the end of the chapter.

Methodology

The proposed pipeline of the offensive language detection system has four main phases. During the first phase, several dialectal Arabic datasets that cover various types of offensive content from user-generated content platforms are collected, binarized the labels to have only offensive and not offensive classes, and classified into training and testing datasets. After that, the training datasets from all datasets are used to fine-tune a customized version of AraBERT that have been pre-trained further to add more dialectal and cultural knowledge to the task of Arabic offensive language detection. Outputs from the customized AraBERT model are used as input to the classification model to train the model. In the last phase, performance evaluation procedures are conducted to examine

the predictions from the classification model. I test the significance of performance differences using the Paired Bootstrap test provided by Dror et al. (2018) and described in Chapter 3 at p-value of 0.05. Figure 1 shows the proposed system pipeline.

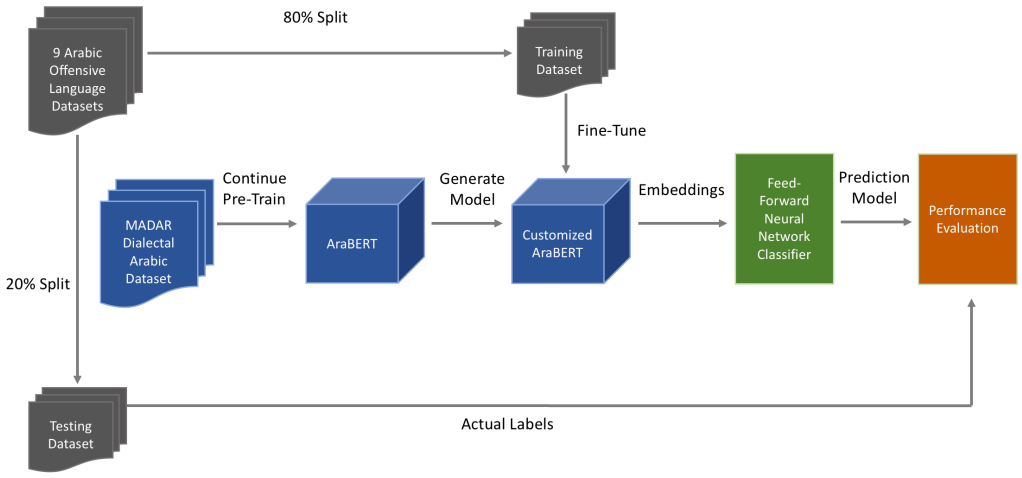


Figure 45 Proposed pipeline for Arabic offensive language detection system based on dialectal continued pre-trained model

Datasets

The nine Arabic offensive language datasets that were analyzed in Chapter 5 and reintroduced in Chapter 6 are adopted to develop the proposed classification system in this chapter. The datasets consist of Aljazeera.net Deleted Comments (Mubarak, Darwish, & Magdy, 2017), Egyptian Tweets (Mubarak, Darwish, & Magdy, 2017), YouTube Comments (Alakrot, Murray, & Nikolov, 2018), Religious Hate Speech (Albadi, Kurdi, & Mishra, 2018), L-HSAB (Mulki et al., 2019), T-HSAB (Haddad et al., 2019), MPOLD (Chowdhury et al., 2020), OSACT (Mubarak et al., 2020), and the Multi-

Platform Hate Speech Dataset (Omar et al., 2020). All datasets were used without any changes in content. However, to maintain consistency among all datasets, the labels were changed for some datasets. Thus, only two classes are maintained; offensive and not offensive. Datasets are also randomly classified into 80% training dataset and 20% testing dataset. Table 37 presents the distribution of each dataset after performing train-test split.

Table 37 Datasets Distribution

Dataset	Train	Test
Aljazeera.net Deleted Comments	25,353 comments (not offensive = 4,528, offensive = 20,825)	6,339 comments (not offensive = 1,125, offensive = 5,214)
Egyptian Tweets	880 tweets (not offensive = 353, offensive = 527)	220 tweets (not offensive = 100, offensive = 120)
YouTube Comments	12,040 comments (not offensive = 7,379, offensive = 4,661)	3,010 comments (not offensive = 1,858, offensive = 1,152)
Religious Hate Speech	4,909 tweets (not offensive = 2,726, offensive = 2,183)	1,228 tweets (not offensive = 649, offensive = 579)
T-HSAB	4,819 samples (not offensive = 3,068, offensive = 1,751)	1,205 samples (not offensive = 752, offensive = 453)
L-HSAB	4676 tweets (not offensive = 2,919, offensive = 1,757)	1,170 tweets (not offensive = 731, offensive = 439)
MPOLD	3,200 samples (not offensive = 2,660, offensive = 540)	800 samples (not offensive = 665, offensive = 135)
OSACT	8,000 tweets (not offensive = 6,403, offensive = 1,597)	2,000 tweets (not offensive = 1,606, offensive = 394)
Multi-Platform Hate Speech Dataset	16,004 samples (not offensive = 7,973, offensive = 8,031)	4,001 samples (not offensive = 2,027, offensive = 1,974)

The AraBERT Model

Similar to the previous chapters, AraBERT model forms the basic building block of the experiment conducted in this chapter. The model is deployed through the Huggingface library.

Continue Pre-Training (Customized AraBERT)

I apply the MLM to continue pre-train AraBERT as described earlier in Chapter 8. The AraBERT model is trained using MSA corpus that is mostly extracted from Arabic news. Thus, dialectal text in user-generated content format is not captured during the original pre-training process. For this reason, a dialectal corpus is used to continue pre-training the AraBERT model. This continue pre-training process solves out-of-vocabulary issue and updates all weights to achieve a better contextualized word embedding covering the new added vocabulary tokens. The MADAR corpus from the Shared Task on Arabic Fine-Grained Dialect Identification is applied during the continued training process (Salameh, Bouamor, & Habash, 2018; Bouamor et al., 2018; Bouamor, Hassan, & Habash. 2019). The MADAR corpus has two parts as follow:

1. The MADAR Travel Domain Dialect Identification Subtask:

This travel domain-specific dataset has two sub-datasets; the first one consists of 2,000 parallel sentences from 25 Arabic regions in addition to MSA. The 25 regions include the following:

- Gulf of Aden (Yemen (Sana'a), Djibouti, Somalia)
- Gulf (Oman (Muscat), United Arab Emirates, Qatar (Doha), Bahrain, Kuwait, Saudi Arabia (Riyadh, Jeddah), Iraq (Baghdad, Mosul, Basra))

- Levant (Syria (Damascus, Aleppo), Lebanon (Beirut), Jordan (Amman, Salt), Palestine (Al-Quds))
- Nile Basin (Egypt (Cairo, Alexandria, Aswan), Sudan (Khartoum))
- Maghreb (Libya (Tripoli, Benghazi), Tunisia (Tunis, Sfax), Algeria (Algiers), Morocco (Rabat, Fes), Mauritania)

The second part of the dataset consists of additional 10,000 sentences to the previous 2,000 sentences. These sentences are labeled into five Arabic cities (Beirut, Cairo, Doha, Tunis, and Rabat) and MSA.

2. The MADAR Twitter User Dialect Identification Subtask:

This dataset consists of 2,980 Twitter user profiles from 21 different Arabic countries, as shown in Table 38. As can be noticed from the table, the majority of the content is in the Saudi dialect.

Table 38 Distribution of the tweet Profiles by the country label in the MADAR Twitter Corpus (Bouamor, Hassan, & Habash. 2019, p.4)

Country	Count	Percentage
Saudi Arabia	1,070	35.91
Kuwait	213	7.15
Egypt	173	5.81
UAE	152	5.10
Oman	138	4.63
Yemen	136	4.56
Qatar	126	4.22
Bahrain	113	3.79
Jordan	107	3.59
Sudan	100	3.36
Iraq	99	3.32
Algeria	92	3.09
Libya	78	2.62
Palestine	74	2.48
Lebanon	66	2.21
Somalia	60	2.01
Tunisia	51	1.71
Syria	48	1.61
Morocco	45	1.51
Mauritania	37	1.24
Djibouti	2	0.07
Comoros	0	0
Total Annotated	**2,980**	**100**

Each of the datasets is provided in three parts; train, development, and test. All parts of both MADAR datasets were used to continue pre-training AraBERT, excluding labels, to ensure the coverage of most Arabic dialects, after removing duplications and without cleaning or filtering any content. Text was segmented using the Farasa segmenter

tool and tokenized using wordPiece tokenizer, similar to the process followed in training the original AraBERT model.

Classification Model

The pooled output from customized AraBERT encoder is used with a simple FFNN layer to build the classification model. Similar experimental settings of those previously applied during chapters 8 and 9 are used in this chapter. All experiments are created using the same environment, tools, and classifier from the previous chapters (Chapters 8 and 9).

Performance Evaluation

All experiments were evaluated with macro-averaged precision, recall, F1, and accuracy. Evaluation per class is considered as well.

Results

Table 39 reports precision, recall, F1, and accuracy scores per class and macro-averaged over all classes in each dataset. For some datasets that have been used by previous studies, the best reported F1 score is included in Table 39 along with their classification model as SOTA score. Having the SOTA information supports the comparison between results obtained from the customized AraBERT model and previous best classification models. To further evaluate the effect from the customized AraBERT, I apply the same pipeline described in Figure 45 Proposed pipeline for Arabic offensive language detection system based on dialectal continued pre-trained model to the original AraBERT model and report the results.

Table 39 Performance results

Test Dataset	Customized AraBERT					AraBERT F1	SOTA F1/ Model	Significant Test*
	Label	Precision	Recall	F1	Accuracy			
Aljazeera.net Deleted Comments	Not	0.57	0.54	0.55	0.85	0.73	0.40/ SVM (Chowdhury et al., 2020)	0.870
	Off	0.90	0.91	0.91				
	Macro	0.73	0.72	**0.73**				
Egyptian Tweets	Not	0.71	0.77	0.74	0.75	0.69	0.90/ FastText (Mubarak & Darwish, 2019)	0.978
	Off	0.79	0.74	0.77				
	Macro	0.75	0.76	**0.75**				
Religious Hate Speech	Not	0.77	0.80	0.79	0.80	0.79	0.77/ GRU (Albadi, Kurdi, & Mishra, 2018)	0.000
	Off	0.82	0.79	0.80				
	Macro	0.80	0.80	**0.80**				
YouTube Comments	Not	0.82	0.85	0.83	0.87	0.86	0.82/ SVM (Chowdhury et al., 2020)	0.052
	Off	0.90	0.89	0.89				
	Macro	0.86	0.87	**0.86**				
L-HSAB	Not	0.73	0.89	0.80	0.83	0.84	0.896/ NB (Mulki et al., 2019)	0.000
	Off	0.92	0.80	0.86				
	Macro	0.82	0.84	**0.83**				
T-HSAB	Not	0.86	0.77	0.81	0.78	0.78	0.923/ NB (Haddad, Mulki, & Oueslati, 2019)	0.036
	Off	0.86	0.77	0.81				
	Macro	0.77	0.78	**0.77**				
MPOLD	Not	0.93	0.83	0.88	0.81	0.73	-	0.401
	Off	0.46	0.71	0.56				
	Macro	0.70	0.77	**0.72**				
OSACT	Not	0.98	0.98	0.98	0.96	0.75	0.905/ ensemble method (Hassan et al., 2020)	0.890
	Off	0.67	0.57	0.62				
	Macro	0.83	0.78	**0.80**				
Multi-platform Hate Speech	Not	1.00	0.96	0.98	0.98	0.98	0.987/ RNN (Omar, Mahmoud, & Abd-El-Hafeez, 2020)	0.061
	Off	0.97	1.00	0.98				
	Macro	0.98	0.98	**0.98**				

* Significant test score is calculated by applying the Paired Bootstrap test at p-value of 0.05 between the original AraBERT model results and results of the continued pre-trained AraBERT model.

Most results do not show significant improvement over the SOTA score, except for Aljazeera.net Deleted Comments, which increases from 0.40 to 0.73 in macro F1 score with significant improvement in results of the offensive class. Some datasets report very close F1 score by the customized AraBERT model to the one reported by SOTA, even though the classification model is not the same, such as the Multi-Platform Hate

Speech. The Egyptian Tweets, L-HSAB, OSACT and T-HSAB datasets record very low macro F1-score in comparison to the SOTA. Previous studies for MPOLD dataset were not found, the customized AraBERT records a macro F1 score of 0.72 with a lower performance for the offensive class in comparison to the not offensive class. Comparing macro-F1 scores from AraBERT with customized AraBERT shows very similar results, except the Egyptian Tweets and the OSACT datasets report significant improvement in performance.

The result from Table 39 demonstrates that for some datasets using traditional machine learning classification models that do not consider the contextual relationship between words reports higher performance score from those that are reported by advanced contextual classification models, such as BERT-based models (AraBERT).

Furthermore, comparing the SOTA results and the proposed system results of the table, the degree to which the cited papers' results differ for some datasets is very significant. There are often multiple other factors that need to be considered when comparing performance between systems, such as differences in the experimental setup. Another conclusion from Table 39 is that the ranking of classification models (e.g., customized AraBERT) ultimately depends on the class and the dataset.

Error Analysis

Table 40 shows examples of misclassified samples from all datasets for the original AraBERT model and the customized AraBERT model.

Table 40 Error analysis for results from AraBERT and customized AraBERT (X: wrong prediction, √ correct prediction)

Dataset	Sample	Label	AraBERT	Customized AraBERT
Aljazeera.net Deleted Comments	باقية وتتمدد باذن الله *Staying and expanding, God willing*	Not offensive	X	X
	انه شهر رمضان شهر الانتصارات نبارك لكم هذه الفتوحات *It is Ramadan, the month of victories. We congratulate you on these conquests*	Not offensive	√	√
	الدولة الإسلامية ستجعلكم تترحمون علي أيام القاعدة و طالبان هته الحرب مختلفة *The Islamic State Will make long for the days of Al Qaeda and the Taliban. This war is different*	Offensive	√	X
	السيسي حارق دم الإخونجية والدواعش وداعميهم *Al-Sisi burned the blood of the Brotherhood, ISIS and their supporters*	Offensive	X	√
Egyptian Tweets	كلب العسكر *Military dog*	Offensive	X	X
	الجاسوس الامريكي وكل اسرائيل موشيه السيسي *The American spy and all of Israel, Al-Sisi*	Offensive	√	√
	هذا الشعب - الا قليلا - يستحق السيسي وما دونه *This people - except for a little - deserves Al-Sisi and less*	Offensive	√	X
	وربنا يلعنك فاكر حلقة امبارح *And our Lord curse you, I think yesterday's episode*	Not offensive	X	√
Religious Hate Speech	ولن ترضى عنك اليهود والنصارى *The Jews and the Christians will never satisfied by you*	Not offensive	X	X
	مقطع من مشاركة الخطيب الشيخ محمود جاسم الكراني في محاضرة تحت عنوان | التوحيد دليل وهم الإلحاد...| *Excerpt from the participation of the Sheikh Mahmoud Jasim Al-Karani in a lecture under the title | Monotheism is a guide to the illusion of atheism | ...*	Offensive	√	√
	الجزائر عربية سنية يجب على أهلها الإبتعاد عن الشيعة المجوس فوالله ما دخلوا دولة إلا وعاثوا فيها فسادا وأقرب دليل اليمن والعراق وسوريا #حاسبوا_امير_موسوي *Algeria is a Sunni Arab and its people must stay away from the Shiites. By God, they did not enter a country but wreaked havoc in it. The closest examples are Yemen, Iraq and Syria.*	Offensive	√	X
	اللبنانيّون الشيعة و«العونيّون» على لوائح الترحيل من السعوديّة *Lebanese Shiites and "Aounis" on the list of deportation from Saudi Arabia*	Not offensive	X	√
YouTube Comments	هههههههه قوي *Hahahahahaha strong*	Not offensive	X	X
	عيب كبير راغب علامة كبير عليك يا مجنونة راغب علامة ستار *Shame on you, Ragheb Allamah is big to you, you are a crazy of Ragheb Allamah star*	Offensive	√	√
	اعطوني إياها بس ثاني يوم ما راح تشوفها ولك.....جلطتيني	Offensive	√	X

	بنت الكلب *They gave it (or her) back to me but I promise they won't see it (or her) from now on. She gave me a heart attack that daughter of a dog*			
	ماعرفناج خليجيه لو لبنانيه *Are you from the Gulf or Lebanon?*	Not offensive	X	√
L-HSAB	لحكي موجه لجحاش اربعطعش الشهر وكل شي عدا ذلك وخاصة أهل المقاومة عراسي من فوق *My talk is directed to an animal of fourteenth of the month and everything else especially people of the resistance are on top of my head*	Offensive	X	X
	كلنا ثقة فيك ايها القائد *We all have confidence in you, Commander*	Not offensive	√	√
	ليش بتكرهيه جبران باسيل أصدق واحد في لبنان والكذابين أكيد مش راح يحبوه الغيرة قاسية على صاحبها *Why do you hate Gebran Bassil? He is the most trusted one in Lebanon, and the liars for sure will not love him. Jealousy is cruel to them*	Offensive	√	X
	بصراحه معنا حق اسألت البنات بنص الماتش مستفزي و غبية *Frankly, we have the right to ask the girls in the text of the match, which is provocative and stupid*	Offensive	X	√
T-HSAB	شتان العالم الجاهل *The diversions of the ignorant world*	Not offensive	X	X
	شكرا ويحفظكم *Thanks and bless you*	Not offensive	√	√
	مايحدث سياسة ممنهجة ومدروسة وأثارها بعيدة المدى *What is happening is a systematic and deliberate policy and its effects are far-reaching*	Not offensive	√	X
	اليهود اكذب الناس والعن الناس تريد الخيانه صاحب يهودي *Jews are the biggest liars amongst all people. If you want to get betrayed then befriend a Jew*	Offensive	X	√
MPOLD	الحمد لله الذي أخرس هذا الزنيق بنحره كالخروف *Praise be to God who silenced this heretic by slaughtering him like a lamb*	Offensive	X	X
	لعن الله عليكم يا اخوان الشياطين God curse you, brothers of evils	Offensive	√	√
	الله يقويك ياترامب على هالكلاب *May God strengthen you, Trump, against these dogs*	Offensive	√	X
	دي بني ادمة مريضة اصلا مش عارفينلها ملة *This women is already sick, and no one know her*	Offensive	X	√
OSACT	@USER @USER قال شف وجه العنز واحلب لبن *@USER @USER said, See the face of goats and milk them*	Offensive	X	X
	يا حُب التملك يا طبعي الأناني. *Oh my love of possession, my selfish nature.*	Not offensive	√	√
	يا نموت ثوار يا نعيش أحرار *Either we die revolutionaries or we live free*	Not offensive	√	X
	@USER أيوه إنت يا حماده يا حبيبي تبطل فتي😏 *@USER Oh, oh my beloved, Hamadah, stop saying things* 😏	Not offensive	X	√

Multi-platform Hate Speech	وايش بعد هههههههه ليتك ماركزت عليها *What else left Hahahahahahahahahahah, I wish you did not focus on her*	Not offensive	X	X
	مايحتاج تقول اسم زوجتك في كل المقاطع غثيتنا تراك *You don't need to say your wife's name in all the clips*	Not offensive	√	√
	نتمنى ان لا تستهدف الشعب المكلوم ولا @User.IDX البنية التحتية للفلسطينيين الشرفاء واذا بترد فعليها باستهداف اذناب قطر وأيران ولاكن نعلم ان الخونة في عيون نتنياهو وحكومته وتحت حمايتهم رحم الله الشهيد البطل احمد ياسين *@ User.IDX We hope that it does not target the grieving people or the infrastructure of the honorable Palestinians, and if it responds, it should target the offspring of Qatar and Iran, but we know that the traitors are in the eyes of Netanyahu and his government and under their protection, may God have mercy on the hero martyr Ahmed Yassin*	Not offensive	√	X
	مهما يكبر همك! الله اكبر *Whatever grows your concern! Allah is the greatest*	Not offensive	X	√

I manually inspected the misclassified tweets to search for overlapping features or patterns and identify noise sources, confusing the classifier. Some of the patterns can be summarized in the following points:

- Names of persons or groups of people. For example, "ترامب/ Trump", " أهل المقاومة / people of the resistance", and "باسيل جبران / Bassil Gebran".
- Short samples, especially short offensive samples, as they show less context and make it hard for the classifier to understand correctly. For example, "كلب العسكر/ military dog" and "شتان العالم الجاهل / the diversions of the ignorant world".
- The term "هههههه / hahahah" in both offensive and not offensive samples.
- Animal names are frequently used in offensive samples, specifically in hate speech class. For instance, "كلاب / dogs" and "خروف/ lamb".

Chapter Summary

Dialectal and cultural-based offensive language detection is covered in this chapter. The proposed method includes dataset preparation, fine-tuning with nine datasets, generating contextual embeddings by a customized AraBERT model, applying a FFNN layer to predict the class. The evaluation includes statistical metrics. The results report the need to add more dialectal data for the continuing pre-training process to create noticeable effects on the model's performance over those obtained from the original AraBERT model.

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

Amir al-Mu'minin Imām ʻAlī ibn Abī Tālib (peace and mercy be upon him) said:
" من أفحش شفا حساده."
“One who uses obscene language cures [the envy of] his enviers [and brings them joy].”

- Ghurar al-Hikam, number 1508; Mizan ul Hikmah, page 1252

In this dissertation, I present methods for automatically detecting Arabic offensive language from user-generated content. The main goal is to develop a system for detecting Arabic offensive language that incorporates cultural and dialectal knowledge, which reduces the ambiguity of having several heterogeneous Arabic dialects with various sub-cultures.

Findings from the experiments conducted during this dissertation can be summarized by addressing the research hypotheses that are defined in chapter 1, as follows:

H1. Applying a contextual-aware pre-processing approach in cleaning and normalizing text increases the performance of the classifier.

In chapter 7, I examine the impact of several pre-processing steps: conversion of emojis to textual labels, hashtags segmentation, normalization of different forms of

Arabic letters, normalization of selected nouns from Arabic dialects to MSA, conversion of selected hyponyms to hypernyms, and basic cleaning processes such as removing numbers, Kashida, diacritics, and HTML tags. The experiments cover several classifiers, including traditional machine learning classifiers, ensemble machine learning classifiers, deep learning classifiers with BOW and AraVec, and BERT-based classifiers. The results show unequal effects of pre-processing steps based on the classifier and demonstrate the limited outcomes from Arabic text pre-processing on offensive language detection. The advanced BERT-based classifiers show very limited improvements from pre-processing text and justify omitting this step for text classification.

H2. Creating a dialectal representation reduces inconsistency among dialects; therefore, boosts the performance of the classifier.

Chapter 8 evaluates the impact of cross-dialect transfer learning using AraBERT model for three Arabic dialects; Levantine, Tunisian, and Egyptian. Results demonstrate significant variations in the impact of transferring linguistic knowledge among dialects. While the Levantine dialect shows the highest performance score among the three dialects, cross-dialectal transfer learning reduces its performance. In contrast, Egyptian and Tunisian dialects record significant improvements.

H3. Developing a SOTA classifier based on a dialectal representation increases the accuracy of detecting offensive language in multi-dialectal corpus and dialectal specific corpus.

This hypothesis is tested along several dimensions using the SOTA BERT model; dialect, platform, and cultural and dialectal. The discussion of H2 above from chapter 8 also applied to test this hypothesis from a dialectal vs. multi-dialectal perspective. Chapter 9 also depends on AraBERT, as it applies cross-platform transfer learning using several Arabic offensive detection datasets from multiple platforms; Twitter, YouTube, and Aljazeera. The results report an overall reduction in performance when applying transfer learning across individual datasets from heterogeneous platform's sources and domains (social media, news).

Limitations

All experiments in this dissertation were conducted using the Colab Google Pro environment, which offers limited RAM capacity affecting the process of continuing pre-training the BERT model. I consider a medium size dialectal corpus during continuing BERT training to avoid run time crashes caused by the limited capacity.

Datasets used throughout this dissertation's work are not constructed by one entity. Rather they have been gathered from several researchers with different annotating and filtering procedures. Thus, consistency is not maintained among the datasets. Besides, none of the datasets that were publicly available online is in Gulf dialect, which might affect the representativeness of Arabic dialects.

Future Work

Various extensions of the research conducted during this dissertation are worthy of future research. It is important to expand the proposed customized AraBERT model for Arabic offensive language detection from multiple aspects.

Improving the vocabulary list of AraBERT to add domain specific terms that are very frequent and valuable for offensive language detection, is one possible way to improve the customized model.

Furthermore, continuing pre-training the customized AraBERT model using large dialectal datasets, that have been extracted from social media platforms using locations features rather than topical content, is also crucial to consider for future studies as that can support the learning process of sub-cultures.

Another significant extension of this dissertation is concerned with applying more advanced network architectures such as CNN, Bi-LSTM, or CNN-LSTM on top of the customized AraBERT model's encoder or applying some methods that depend on ensemble approaches.

While some models that have been explored in this dissertation report low improvement in performance, developing an ensemble method that aggregate the results among all models could be beneficial for offensive language detection, and a great step to extend the work of this dissertation.

REFERENCES

Abdelfatah, K., Terejanu, G., & and Alhelbawy, A. (2017). Unsupervised Detection of Violent Content in Arabic Social Media. *Comput. Sci. Inf. Technol*. (CS IT) (pp. 1–7).

Abdul-Mageed, M., Elmadany, A., & Nagoudi, E. M. B. (2020). ARBERT & MARBERT: Deep Bidirectional Transformers for Arabic. *arXiv preprint arXiv:2101.01785*.

Abozinadah, E. , Mbaziira, A., & Jones, J. (2015). Detection of Abusive Accounts with Arabic Tweets. *International Journal of Knowledge Engineering*, 113-119, Vol. 1, No. 2. DOI: 10.7763/IJKE.2015.V1.19

Abozinadah, E. (2017). Detecting Abusive Arabic Language Twitter Accounts Using a Multidimensional Analysis Model (Doctoral dissertation).

Abozinadah, E. A. (2016). Improved micro-blog classification for detecting abusive Arabic Twitter accounts. *International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol*, *6*.

Abozinadah, E., & Jones Jr, J. (2017, May). A Statistical Learning Approach to Detect Abusive Twitter Accounts*. In Proceedings of the International Conference on Compute and Data Analysis (ICCDA '17)* (pp. 6-13). Association for Computing Machinery, New York, NY, USA. DOI:https://doi.org/10.1145/3093241.3093281

Abu Farha, I. A., & Magdy, W. (2019, August). Mazajak: An Online Arabic Sentiment Analyser. *Proceedings of the Fourth Arabic Natural Language Processing Workshop* (pp. 192-198). Association for Computational Linguistics (ACL). Florence, Italy. DOI: 10.18653/v1/W19-4621.

Abu Farha, I. A., & Magdy, W. (2020, May). From Arabic Sentiment Analysis to Sarcasm Detection: The ArSarcasm Dataset. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 32-39). Language Resources and Evaluation Conference (LREC 2020), Marseille, France, 11–16 May 2020.

Abu Farha, I., & Magdy, W. (2020). Multitask Learning for Arabic Offensive Language and Hate-Speech Detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 86–90). European Language Resource Association.

Abu Farha, I., Zaghouani, W., & Magdy, W. (2021). Overview of the WANLP 2021 Shared Task on Sarcasm and Sentiment Detection in Arabic. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*.

Al-Hassan, A., & Al-Dossari, H. (2019, February). Detection of Hate Speech in Social Networks: a Survey on Multilingual Corpus. In *6th International Conference on Computer Science and Information Technology* (Vol. 10).

Alakrot, A., Murray, L., & Nikolov, N. S. (2018a). Dataset Construction for the Detection of Anti-Social Behaviour in Online Communication in Arabic. *Procedia Computer Science*, *142*, 174–181. https://doi.org/https://doi.org/10.1016/j.procs.2018.10.473

Alakrot, A., Murray, L., & Nikolov, N. S. (2018b). Towards Accurate Detection of Offensive Language in Online Communication in Arabic. *Procedia Computer Science*, *142*, 315–320. https://doi.org/https://doi.org/10.1016/j.procs.2018.10.491

Albadi, N., Kurdi, M., & Mishra, S. (2018). Are they Our Brothers? Analysis and Detection of Religious Hate Speech in the Arabic Twittersphere. *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (pp.69–76). https://doi.org/10.1109/ASONAM.2018.8508247

Albadi, N., Kurdi, M., & Mishra, S. (2019). Hateful People or Hateful Bots?: Detection and Characterization of Bots Spreading Religious Hatred in Arabic Social Media. *Proceedings of the ACM on Human-Computer Interaction*, *3*(CSCW) (pp.1–25). https://doi.org/10.1145/3359163

Alharbi, A., & Lee, M. (2020). Combining Character and Word Embeddings for the Detection of Offensive Language in Arabic. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 91–96). European Language Resource Association.

Alhumoud, S., Altuwaijri, M., Albuhairi, T., & Alohaideb, W. (2015). Survey on Arabic Sentiment Analysis in Twitter. *International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering,* 9 (1): 364-378.

Aljarah, I., Habib, M., Hijazi, N., Faris, H., Qaddoura, R., Hammo, B., Abushariah, M., & Alfawareh, M. (2020). Intelligent Detection of Hate Speech in Arabic Social Network: A Machine Learning Approach. *Journal of Information Science*, 0165551520917651. SAGE Publications Sage UK: London, England

Alrefaie, M. (2016). Arabic-stop-words [Github Repository]. Retrieved on January 24, 2020 from https://github.com/mohataher/arabic-stop-words

Alshehri, A., El Moatez Billah Nagoudi, H. A., & Abdul-Mageed, M. (2018, May). Think before Your Click: Data and Models for Adult Content in Arabic Twitter. In *TA-COS 2018: 2nd Workshop on Text Analytics for Cybersecurity and Online Safety* (p. 15).

ALW (2019). ALW3: 3rd Workshop on Abusive Language Online. Retrieved on January 19, 2020 from https://sites.google.com/view/alw3/home

Amidi, A., & Amidi, S. (n.d.). CS 230: Deep Learning: Winter 2019 [Class Cheatcheets]. Retrieved from https://stanford.edu/~shervine/teaching/

Antoun, W., Baly, F., & Hajj, H. (2020). AraBERT: Transformer-based Model for Arabic Language Understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 9–15). European Language Resource Association.

Arab Youth Survey. (n.d.). ASDA'A BCW Report 2020. Retrieved March 7, 2021, from
Azure, P. (2021). Transfer Learning for Natural Language Processing. ISBN 9781617297267.
Basile, V., Bosco, C., Fersini, E., Debora, N., Patti, V., Pardo, F. M. R., ... & Sanguinetti, M. (2019). Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation* (pp. 54-63). Association for Computational Linguistics.
Bohra, A., Vijay, D., Singh, V., Akhtar, S. S., & Shrivastava, M. (2018, June). A Dataset of Hindi-English Code-mixed Social Media Text for Hate Speech Detection. In *Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media* (pp. 36-41).
Bosco, C., Felice, D. O., Poletto, F., Sanguinetti, M., & Maurizio, T. (2018). Overview of the EVALITA 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian* (Vol. 2263, pp. 1-9). CEUR.
Botts, C. (2019) Evaluation metrics in NLP [Lecture note]. Retrieved from https://nbviewer.jupyter.org/github/cgpotts/cs224u/blob/2019-spring/evaluation_metrics.ipynb
Bouamor, H., Habash, N., Salameh, M., Zaghouani, W., Rambow, O., Abdulrahim, D., Obeid, O., Khalifa, S., Eryani, F., Erdmann, A., & Oflazer, K. (2018). The MADAR Arabic Dialect Corpus and Lexicon. Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA).Miyazaki, Japan, 7-12 May 2018.
Bouamor, H., Hassan, S., & Habash, N. (2019). The MADAR Shared Task on Arabic Fine-Grained Dialect Identification. Proceedings of the Fourth Arabic Natural Language Processing Workshop (WANLP19) (pp. 199–207). Association for Computational Linguistics (ACL). Florence, Italy.
Boudad, N., Faizi, R., Thami, R., & Chiheb, R. (2018). Sentiment analysis in Arabic: A review of the literature. *Ain Shams Engineering Journal*, 9(4), 2479–2490. https://doi.org/10.1016/j.asej.2017.04.007
Chowdhury, A. G., Didolkar, A., Sawhney, R., & Shah, R. (2019, July). ARHNet-leveraging community interaction for detection of religious hate speech in arabic. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop* (pp. 273-280). Association for Computational Linguistics (ACL). Florence, Italy. DOI: 10.18653/v1/P19-2038
Chowdhury, S., Mubarak, H., Abdelali, A., Jung, B., & Salminen, J. (2020). A Multi-Platform Arabic News Comment Dataset for Offensive Language Detection. In *Proceedings of The 12th Language Resources and Evaluation Conference* (pp. 6203–6212). European Language Resources Association.
Çöltekin, Ç. (2020, May). A Corpus of Turkish Offensive Language on Social Media. *Proceedings of the 12th Language Resources and Evaluation Conference* (pp. 6174-6184). European Language Resources Association. Marseille, France.

Dadvar, M., Jong, F. D., Ordelman, R., & Trieschnigg, D. (2012). Improved Cyberbullying Detection Using Gender Information. *Proceedings of the Twelfth Dutch-Belgian Information Retrieval Workshop (DIR 2012)*. University of Ghent.

Dadvar, M., Trieschnigg, R., Ordelman, R., de Jong, F., (2013) Improving Cyberbullying Detection with User Context. *Proceedings of the 35th European Conference on IR Research (ECIR) (pp.*693–696).

Dahou, A., Elaziz, M. A., Zhou, J., & Xiong, S. (2019). Arabic Sentiment Classification Using Convolutional Neural Network and Differential Evolution Algorithm. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2019/2537689

Darwish, K. (2013, October). Arabizi Detection and Conversion to Arabic. Proceedings of the (EMNLP) 2014 Workshop on Arabic Natural Language Processing (ANLP) (pp. 217-224). Association for Computational Linguistics (ACL). Doha, Qatar.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017, May). Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media* (Vol. 11, No. 1).

Derczynski, L. (2019). Simple Natural Language Processing Tools for Danish. https://arxiv.org/pdf/1906.11608.pdf

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

Djandji, M., Baly, F., & Hajj, H. (2020, May). Multi-Task Learning Using AraBERT for Offensive Language Detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 97-101).

Dror, R., Baumer, G., Shlomov, S., & Reichart, R. (2018). The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1383–1392). Association for Computational Linguistics.

Eisenstein, J. (2018) Natural Language Processing. MIT press. Retrieved from https://github.com/jacobeisenstein/gt-nlp-class/blob/master/notes/eisenstein-nlp-notes.pdf

El Gayar, N., & Suen, C. (2018). Computational Linguistics, Speech and Image Processing for Arabic Language. *Series on Language Processing, Pattern Recognition, and Intelligent Systems,* 7. Doi.org/10.1142/10693

ElJundi, O., Antoun, W., El Droubi, N., Hajj, H., El-Hajj, W., & Shaban, K. (2019). hULMonA: The Universal Language Model in Arabic. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop* (pp. 68–77). Association for Computational Linguistics.

Elmadany, A., Zhang, C., Abdul-Mageed, M., & Hashemi, A. (2020). Leveraging Affective Bidirectional Transformers for Offensive Language Detection. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and*

Processing Tools, with a Shared Task on Offensive Language Detection (pp. 102–108). European Language Resource Association.

Facebook Community Standards (n.d.). 12. Hate Speech. Retrieved on January 19, 2020 from https://www.facebook.com/communitystandards/hate_speech

Fersini, E., Nozza, D., & Rosso, P. (2018). Overview of the evalita 2018 task on automatic misogyny identification (ami). *EVALITA Evaluation of NLP and Speech Tools for Italian*, *12*, 59.

Fersini, E., Nozza, D., & Rosso, P. (2020). Ami@ evalita2020: Automatic misogyny identification. *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020), Online. CEUR. org*.

Fortuna, P. & Nunes, S. (2018). A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Survy, 51(4).*

Founta, A., Chatzakou, D., Kourtellis, N., Blackburn, J., Vakali, A., & Leontiadis, I. (2018). A Unified Deep Learning Architecture for Abuse Detection. *Proceedings of the 10th ACM Conference on Web Science (WebSci '19)(pp.105–114).* Association for Computing Machinery, New York, NY, USA. DOI: https://doi.org/10.1145/3292522.3326028

Gablasova, D., Brezina, V., & McEnery, T. (2017). Collocations in corpus-based language learning research: Identifying, comparing, and interpreting the evidence. *Language learning*, *67*(S1), 155-179.

Gitari, N., Zhang, Z., Zhang, Z., Damien, H., Long, J. (2015). A Lexicon-based Approach for Hate Speech Detection. *International Journal of Multimedia and Ubiquitous Engineering*, *10*, 215–230. https://doi.org/10.14257/ijmue.2015.10.4.21

Goldberg, Y. (2015, October). A Primer on Neural Network Models for Natural Language Processing [draft]. Retrieved from http://u.cs.biu.ac.il/~yogo/nnlp.pdf.

Goodfellow, I.; Bengio, B.; Courville, A. (2016). Deep Learning. MIT press. Retrieved from http://www.deeplearningbook.org/contents/ml.html

Habash, N. (2010). Introduction to Arabic Natural Language Processing. *Synthesis Lectures on Human Language Technologies*, *3*(1), 1-187. Morgan & Claypool Publishers.

HaCohen-Kerner, Y., Miller, D., & Yigal, Y. (2020). The influence of preprocessing on text classification using a bag-of-words representation. *PloS one*, *15*(5), e0232525. https://doi.org/10.1371/journal.pone.0232525

Haddad, B., Orabe, Z., Al-Abood, A., & Ghneim, N. (2020). Arabic Offensive Language Detection with Attention-based Deep Neural Networks. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 76–81). European Language Resource Association.

Haddad, H., Mulki, H., & Oueslati, A. (2019, October). T-HSAB: A Tunisian Hate Speech and Abusive Dataset. In *International Conference on Arabic Language Processing* (pp. 251-263). Springer, Cham.

Haidar, B., Chamoun, M., & Serhrouchni, A. (2017, October). Multilingual Cyberbullying Detection System: Detecting Cyberbullying in Arabic Content.

In *2017 1st Cyber Security in Networking Conference (CSNet)* (pp. 1-8). Rio de Janeiro, Brazil, 18-20 October 2017. DOI: 10.1109/CSNET.2017.8242005.

Haidar, B., Chamoun, M., & Serhrouchni, A. (2018). Arabic Cyberbullying Detection: Using Deep Learning. *2018 7th International Conference on Computer and Communication Engineering (ICCCE)* (pp.284–289). https://doi.org/10.1109/ICCCE.2018.8539303

Haidar, B., Chamoun, M., & Serhrouchni, A. (2019). Arabic Cyberbullying Detection: Enhancing Performance by Using Ensemble Machine Learning. *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData),* (pp.323–327). https://doi.org/10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00074

Hassan, S., Samih, Y., Mubarak, H., Abdelali, A., Rashed,A., & Absar Chowdhury, S. (2020, May). ALT Submission for OSACT Shared Task on Offensive Language Detection. *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 61-65). Language Resources and Evaluation Conference (LREC 2020), Marseille, France, 11–16 May 2020.

Hatebase (n.d.). "The world's largest structured repository of regionalized, multilingual hate speech". Retrieved from https://hatebase.org/ .

Hijjawi, M., & and Elsheikh, Y. (2015). *A*rabic Language Challenges in Text Based Conversational Agents Compared to The English Language. *International Journal of Computer Science and Information Technology*, 7 (3). Doi.org/10.5121/ijcsit.2015.7301

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M. & Gelly, S. (2019). Parameter-efficient Transfer Learning for NLP. In *International Conference on Machine Learning* (pp. 2790-2799). PMLR.

https://www.arabyouthsurvey.com/

Husain, F. (2020). Arabic Offensive Language Detection Using Machine Learning and Ensemble Machine Learning Approaches. *arXiv preprint* arXiv: 2005.08946.

Husain, F. (2020). OSACT4 Shared Task on Offensive Language Detection: Intensive Preprocessing-Based Approach. *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 53–60). Language Resources and Evaluation Conference (LREC 2020), Marseille, France, 11–16 May 2020.

Husain, F. & Uzuner, O. (2021). A Survey of Offensive Language Detection for the Arabic Language. The ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP).

Husain, F. & Uzuner, O. (2021). Leveraging Offensive Language for Sarcasm and Sentiment Detection in Arabic. *Proceedings of the Sixth Arabic Natural Language Processing Workshop*. Association for Computational Linguistics. the 16th conference of the European Chapter of the Association for Computational Linguistics (EACL 2021).

Husain, F. & Uzuner, O. (2021). SalamREPO: an Arabic Offensive Language Knowledge Repository. IEEE International Conference on Computer Applications & Information Security (ICCAIS'2021). Tunisia. 18-20 March 2021.

Husain, F. & Uzuner, O. (2021). Fine-Tuning Approach for Arabic Offensive Language Detection System: BERT-Based Model. IEEE International Conference on Computer Applications & Information Security (ICCAIS'2021). Tunisia. 18-20 March 2021.

Husain, F., & Uzuner, O. (2021). Exploratory Arabic Offensive Language Dataset Analysis. *arXiv preprint arXiv:2101.11434*.

Husain, F., Lee, J., Henry, S., & Uzuner, O. (2020). SalamNET at SemEval-2020 Task 12: Deep Learning Approach for Arabic Offensive Language Detection. *Proceedings of the International Workshop on Semantic Evaluation 2020 (SemEval 2020)*. The 28th International Conference on Computational Linguistic (COLING 2020). Barcelona, Spain, 12-13 December 2020.

Johnston, A., & Weiss, G. (2017). Identifying Sunni Extremist Propaganda with Deep Learning. 2017 IEEE Symposium Series on Computational Intelligence (SSCI) (pp.1–6). https://doi.org/10.1109/SSCI.2017.8280944

Jurafsky, D., & Martin, J. (2009). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. (2nd Edition.) Prentice-Hall.

Jurafsky, D., & Martin, J. (2019). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. (3nd Edition.) draft. Retrieved on January 4, 2019 from https://web.stanford.edu/~jurafsky/slp3/

Kaati, L., Omer, E., Prucha, N., & Shrestha, A. (2015). Detecting Multipliers of Jihadism on Twitter. 2015 IEEE International Conference on Data Mining Workshop (ICDMW) (pp.954–960). https://doi.org/10.1109/ICDMW.2015.9

Kai, M. (2017) Confusion Matrix. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, MA

Kapoor, R., Kumar, Y., Rajput, K., Shah, R. R., Kumaraguru, P., & Zimmermann, R. (2019, July). Mind Your Language: Abuse and Offense Detection for Code-Switched Languages. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 9951-9952. https://doi.org/10.1609/aaai.v33i01.33019951

Keleg, A., El-Beltagy, S., & Khalil, M. (2020). ASU_OPTO at OSACT4 - Offensive Language Detection for Arabic text. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 66–70). European Language Resource Association.

Kent, S. (2018). German Hate Speech Detection on Twitter. *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018).* Vienna, Austria, September 21.

Kolesnikova, O. (2016). Survey of word co-occurrence measures for collocation detection. *Computación y Sistemas*, *20*(3), 327-344.

Kumar, R., Ojha, A. K., Malmasi, S., & Zampieri, M. (2018, August). Benchmarking aggression identification in social media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)* (pp. 1-11).

Kumar, R., Ojha, A. K., Malmasi, S., & Zampieri, M. (2020, May). Evaluating aggression identification in social media. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying* (pp. 1-5).

Kwok, I., & Wang, Y. (2013, June). Locate the Hate: Detecting Tweets Against Blacks. *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence* (pp.1621-1622). Vol. 27, No. 1.

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). Recurrent Convolutional Neural Networks for Text Classification. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (pp. 2267–2273)*. Austin, Texas, USA. AAAI Press.

Lample, G., & Conneau, A. (2019). Cross-lingual Language Model Pretraining. *arXiv preprint arXiv:1901.07291*.

Laub, Z. (Jun.7, 2019). Hate Speech on Social Media: Global Comparison. Council on Foreign Relations. Retrieved from https://www.cfr.org/backgrounder/hate-speech-social-media-global-comparisons

Lees, A., Sorensen, J., & Kivlichan, I. (2020). Jigsaw@ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Bologna, Italy. CEUR. org*.

List of emoticons (n.d.). In Wikipedia. Retrieved January 24, 2020, from https://en.wikipedia.org/wiki/List_of_emoticons

Magdy, W., Darwish, K., & Weber, I. (2015). #FailedRevolutions: Using Twitter to Study the Antecedents of ISIS Support. arXiv preprint arXiv:1503.02401.

Mandl, T., Modha, S., Majumder, P., Patel, D., Dave, M., Mandlia, C., & Patel, A. (2019, December). Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th forum for information retrieval evaluation* (pp. 14-17).

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

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in Pre-Training Distributed Word Representations. *Proceedings of the International Conference on Language Resources and Evaluation*. arXiv:1712.09405

Mironczuk, M. & Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications, 106, 36 - 54.*

Mohaouchane, H., Mourhir, A., & Nikolov, N. (2019, October). Detecting Offensive Language on Arabic Social Media using Deep Learning. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)* (pp. 466-471). IEEE. https://doi.org/10.1109/SNAMS.2019.8931839

Mubarak H., & Darwish K. (2019) Arabic Offensive Language Classification on Twitter. In: Weber I. et al. (eds) Social Informatics. SocInfo 2019. Lecture Notes in Computer Science, vol 11864. Springer, Cham. https://doi-org.mutex.gmu.edu/10.1007/978-3-030-34971-4_18

Mubarak, H., Darwish, K., & Magdy, W. (2017, August). Abusive Language Detection on Arabic Social Media. *Proceedings of the First Workshop on Abusive Language Online* (pp. 52-56). Association for Computational Linguistics (ACL). Vancouver, BC, Canada. https://doi.org/10.18653/v1/w17-3008

Mubarak, H., Darwish, K., Magdy, W., Elsayed, T., & Al-Khalifa, H. (2020, May). Overview of osact4 arabic offensive language detection shared task. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 48-52).

Mulki, H., Haddad, H., Bechikh Ali, C., & Alshabani, H. (2019). L-HSAB: A Levantine Twitter Dataset for Hate Speech and Abusive Language. *Proceedings of the Third Workshop on Abusive Language Online (*pp.111–118*).* Association for Computational Linguistics (ACL). https://doi.org/10.18653/v1/W19-3512

Mustafa, R., Nawaz, M., Ferzund, J., Lali, M., Shahzad, B., & ournier-Viger, P. (2017). Early Detection of Controversial Urdu Speeches from Social Media. Data Science and Pattern Recognition, Ubiquitous International, Volume1, Number2. ISSN 2520-4165

Nabil, M., Aly, M., & Atiya, A. (2015, September). ASTD: Arabic Sentiment Tweets Dataset. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*) (pp. 2515-2519). Association for Computational Linguistics (ACL). Lisbon, Portugal, 17-21 September 2015.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016, April). Abusive Language Detection in Online User Content. *Proceedings of the 25th international conference on world wide web* (pp.145–153). International World Wide Web Conferences Steering Committee.

Omar, A., Mahmoud, T., & Abd-El-Hafeez, T. (2020). Comparative Performance of Machine Learning and Deep Learning Algorithms for Arabic Hate Speech Detection in OSNs. In *Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2020)* (pp. 247–257). Springer International Publishing.

Oprea, S., & Magdy, W. (2019). Exploring Author Context for Detecting Intended vs Perceived Sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 2854–2859). Association for Computational Linguistics.

Orasan, C. (2018). Aggressive Language Identification Using Word Embeddings and Sentiment Features. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)* (pp. 113–119). Association for Computational Linguistics.

Ozel, S., Sarac, E., Akdemir, S., & Aksu, H. (2017). Detection of cyberbullying on social media messages in Turkish. 2017 International Conference on Computer Science and Engineering (UBMK) (pp.366–370). https://doi.org/10.1109/UBMK.2017.8093411

PeaceTech (n.d.). PeaceTech Lab Lexicons. Retrieved on January 19, 2020 from https://www.peacetechlab.org/toolbox-lexicons

Pelicon, A., Martinc, M., & Novak, P. K. (2019, June). Embeddia at SemEval-2019 Task 6: Detecting Hate with Neural Network and Transfer Learning Approaches. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 604-610).

Pennington, J., Socher, R., & Manning, C. (2014, October). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

Pitsilis, G. K., Ramampiaro, H., & Langseth, H. (2018a). Detecting offensive language in tweets using deep learning. *arXiv preprint arXiv:1801.04433*.

Pitsilis, G., Ramampiaro, H., & Langseth, H. (2018b). Effective Hate-Speech Detection in Twitter Data Using Recurrent Neural Networks. *Applied Intelligence*, 48(12), pp. 4730–4742. https://doi.org/10.1007/s10489-018-1242-y

Radcliffe, D., & Abuhmaid, H. (2020). Social Media in the Middle East: 2019 in Review. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3517916

Rehab Duwairi, & Mahmoud El-Orfali (2014). A study of the effects of preprocessing strategies on sentiment analysis for Arabic text. *Journal of Information Science, 40(4), 501-513.*

Ribeiro, M. T., Wu, T., Guestrin, C., & Singh, S. (2020). Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 4902–4912). Association for Computational Linguistics (ACL). 5-10 July 2020.

Rosenthal, S., Farra, N., & Nakov, P. (2017, August). SemEval-2017 Task 4: Sentiment Analysis in Twitter. *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (pp. 502-518). Association for Computational Linguistics (ACL), Vancouver, Canada. DOI: 10.18653/v1/S17-2088

Ross, B., Rist, M., Carbonell, G., Cabrera, B., Kurowsky, N., & Wojatzki, M. (2016). Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC)* (pp. 6-9). Bochumer Linguistische Arbeitsberichte.

Ruder, S. (2019). *Neural Transfer Learning for Natural Language Processing* (Doctoral dissertation, NUI Galway)

Saad, M. (2010). *The Impact of Text Preprocessing and Term Weighting on Arabic Text Classification*. (Doctoral dissertation).

Saeed, H., Calders, T., & Kamiran, F. (2020). OSACT4 Shared Tasks: Ensembled Stacked Classification for Offensive and Hate Speech in Arabic Tweets. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection* (pp. 71–75). European Language Resource Association.

Safaya, A., Abdullatif, M., & Yuret, D. (2020). KUISAIL at SemEval-2020 Task 12: BERT-CNN for Offensive Speech Identification in Social Media. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval)*.

Salameh, M., Bouamor, H., & Habash, N. (2018). Fine- Grained Arabic Dialect Identification. Proceedings of the International Conference on Computational

Linguistics (COLING) (pp.1332-1344). Santa Fe, New Mexico, USA. 20-26 August 2018.

Sanguinetti, M., Comandini, G., Di Nuovo, E., Frenda, S., Stranisci, M., Bosco, C., ... & Russo, I. (2020). HaSpeeDe 2@ EVALITA2020: Overview of the evalita 2020 hate speech detection task. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Online. CEUR. org*.

Sap, M., Card, D., Gabriel, S., Choi, Y., Smith, N. (2019). The Risk of Racial Bias in Hate Speech Detection. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp.1668–1678). Association for Computational Linguistics (ACL), Florence. Italy. https://doi.org/10.18653/v1/P19-1163

Sarkar, D. (2018, April). Implementing Deep Learning Methods and Feature Engineering for Text Data: The Continuous Bag of Words (CBOW) [Blog post]. Retrieved from https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-feature-engineering-text-data-cbow.html

Sarkar, D., Bali, R., & Ghosh, T. (2018). Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras. Packt Publishing Ltd.

Schmidt, A., & Wiegand, M. (2017). Survey on Hate Speech Detection Using Natural Language Processing. *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media (pp.1-10).* Association for Computational Linguistics (ACL). Valencia, Spain. DOI: 10.18653/v1/W17-1101

Sigurbergsson, G. I., & Derczynski, L. (2019). Offensive Language and Hate Speech Detection for Danish. *arXiv preprint arXiv:1908.04531*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., & Potts, C. (2013, October). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp.1631–1642). Association for Computational Linguistics (ACL), Seattle, Washington, USA.

Soliman, A., Eissa, K., & El-Beltagy, S. (2017). AraVec: A Set of Arabic Word Embedding Models for Use in Arabic NLP. *Procedia Computer Science*, 117, 256–265. https://doi.org/10.1016/j.procs.2017.10.117

Su, H., Huang, Z., Chang, H., & Lin, C. (2017). Rephrasing Profanity in Chinese Text. In *Proceedings of the First Workshop on Abusive Language Online* (pp. 18–24). Association for Computational Linguistics.

Uglow, H., Zlocha, M., & Zmyślony, S. (2019). An Exploration of State-of-the-art Methods for Offensive Language Detection. arXiv:1903.07445v2

Unicode Organization (n.d.). Full Emoji List, v13.0. Retrieved from http://www.unicode.org/emoji/charts/full-emoji-list.html

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. In *Proceedings of Recent Advances in Natural Language Processing* (pp. 672-680).

Vandersmissen, B. (2012). Automated detection of offensive language behavior on social networking sites. Ghent University.

Vu, X. S., Vu, T., Tran, M. V., Le-Cong, T., & Nguyen, H. (2020). HSD shared task in VLSP campaign 2019: Hate speech detection for social good. *arXiv preprint arXiv:2007.06493*.

Warner, W., & Hirschberg, J. (2012). Detecting Hate Speech on the World Wide Web. *Proceedings of the Second Workshop on Language in Social Media (LSM '12*) (pp.19–26). Association for Computational Linguistics (*ACL). Montreal, Canada.*

Wiedemann, G., Ruppert, E., Jindal, R., & Biemann, C. (2018). *Transfer Learning from LDA to BiLSTM-CNN for Offensive Language Detection in Twitter*. *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018). arXiv:1811.02906.*

Wiegand, M., Siegel, M., & Ruppenhofer, J. (2018). Overview of the germeval 2018 shared task on the identification of offensive language.

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019a, June). SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). *Proceedings of the 13th International Workshop on Semantic Evaluation (pp.*75-86*)*. Association for Computational Linguistics (*ACL).* Minneapolis, Minnesota, USA. DOI: 10.18653/v1/S19-2010.

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019b). Predicting the Type and Target of Offensive Posts in Social Media. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp.*1415--1420*).* Association for Computational Linguistics (*ACL).* Minneapolis, Minnesota, USA. DOI: 10.18653/v1/N19-1144

Zampieri, M., Nakov, P., Rosenthal, S., Atanasova, P., Karadzhov, G., Mubarak, H., ... & Çöltekin, Ç. (2020). Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020). *arXiv preprint arXiv:2006.07235*.

BIOGRAPHY

Fatemah Ali Husain received her bachelor's degree from Kuwait University in Management Information System. She received a master degree in Information Technology from Kuwait University and a master degree in Computer Science from the University of Minnesota, Twin Cities. Currently, she is a scholar research scientist from the Department of Information Science, College of Life Sciences at Kuwait University. Prior to joining Kuwait University, Fatemah worked in Kuwait at the Ministry of Education, the Supreme Council for Planning and Development, and the Public Authority for Applied Education and Training.